Apostolos Antonacopoulos ·
Subhasis Chaudhuri · Rama Chellappa ·
Cheng-Lin Liu · Saumik Bhattacharya ·
Umapada Pal (Eds.)

LNCS 15319

Pattern Recognition

27th International Conference, ICPR 2024
Kolkata, India, December 1–5, 2024
Proceedings, Part XIX

19 Part XIX

ICPR 2024 INDIA

IAPR

Springer

MOREMEDIA

Lecture Notes in Computer Science 15319

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Apostolos Antonacopoulos ·
Subhasis Chaudhuri · Rama Chellappa ·
Cheng-Lin Liu · Saumik Bhattacharya ·
Umapada Pal
Editors

Pattern Recognition

27th International Conference, ICPR 2024
Kolkata, India, December 1–5, 2024
Proceedings, Part XIX

Springer

Editors

Apostolos Antonacopoulos
University of Salford
Salford, Lancashire, UK

Rama Chellappa
Johns Hopkins University
Baltimore, MD, USA

Saumik Bhattacharya
IIT Kharagpur
Kharagpur, West Bengal, India

Subhasis Chaudhuri
Indian Institute of Technology Bombay
Mumbai, Maharashtra, India

Cheng-Lin Liu
Chinese Academy of Sciences
Beijing, China

Umapada Pal
Indian Statistical Institute Kolkata
Kolkata, West Bengal, India

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-78494-1 ISBN 978-3-031-78495-8 (eBook)
https://doi.org/10.1007/978-3-031-78495-8

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

President's Address

On behalf of the Executive Committee of the International Association for Pattern Recognition (IAPR), I am pleased to welcome you to the 27th International Conference on Pattern Recognition (ICPR 2024), the main scientific event of the IAPR.

After a completely digital ICPR in the middle of the COVID pandemic and the first hybrid version in 2022, we can now enjoy a fully back-to-normal ICPR this year. I look forward to hearing inspirational talks and keynotes, catching up with colleagues during the breaks and making new contacts in an informal way. At the same time, the conference landscape has changed. Hybrid meetings have made their entrance and will continue. It is exciting to experience how this will influence the conference. Planning for a major event like ICPR must take place over a period of several years. This means many decisions had to be made under a cloud of uncertainty, adding to the already large effort needed to produce a successful conference. It is with enormous gratitude, then, that we must thank the team of organizers for their hard work, flexibility, and creativity in organizing this ICPR. ICPR always provides a wonderful opportunity for the community to gather together. I can think of no better location than Kolkata to renew the bonds of our international research community.

Each ICPR is a bit different owing to the vision of its organizing committee. For 2024, the conference has six different tracks reflecting major themes in pattern recognition: Artificial Intelligence, Pattern Recognition and Machine Learning; Computer and Robot Vision; Image, Speech, Signal and Video Processing; Biometrics and Human Computer Interaction; Document Analysis and Recognition; and Biomedical Imaging and Bioinformatics. This reflects the richness of our field. ICPR 2024 also features two dozen workshops, seven tutorials, and 15 competitions; there is something for everyone. Many thanks to those who are leading these activities, which together add significant value to attending ICPR, whether in person or virtually. Because it is important for ICPR to be as accessible as possible to colleagues from all around the world, we are pleased that the IAPR, working with the ICPR organizers, is continuing our practice of awarding travel stipends to a number of early-career authors who demonstrate financial need. Last but not least, we are thankful to the Springer LNCS team for their effort to publish these proceedings.

Among the presentations from distinguished keynote speakers, we are looking forward to the three IAPR Prize Lectures at ICPR 2024. This year we honor the achievements of Tin Kam Ho (IBM Research) with the IAPR's most prestigious King-Sun Fu Prize "for pioneering contributions to multi-classifier systems, random decision forests, and data complexity analysis". The King-Sun Fu Prize is given in recognition of an outstanding technical contribution to the field of pattern recognition. It honors the memory of Professor King-Sun Fu who was instrumental in the founding of IAPR, served as its first president, and is widely recognized for his extensive contributions to the field of pattern recognition.

The Maria Petrou Prize is given to a living female scientist/engineer who has made substantial contributions to the field of Pattern Recognition and whose past contributions, current research activity and future potential may be regarded as a model to both aspiring and established researchers. It honours the memory of Professor Maria Petrou as a scientist of the first rank, and particularly her role as a pioneer for women researchers. This year, the Maria Petrou Prize is given to Guoying Zhao (University of Oulu), "for contributions to video analysis for facial micro-behavior recognition and remote biosignal reading (RPPG) for heart rate analysis and face anti-spoofing".

The J.K. Aggarwal Prize is given to a young scientist who has brought a substantial contribution to a field that is relevant to the IAPR community and whose research work has had a major impact on the field. Professor Aggarwal is widely recognized for his extensive contributions to the field of pattern recognition and for his participation in IAPR's activities. This year, the J.K. Aggarwal Prize goes to Xiaolong Wang (UC San Diego) "for groundbreaking contributions to advancing visual representation learning, utilizing self-supervised and attention-based models to establish fundamental frameworks for creating versatile, general-purpose pattern recognition systems".

During the conference we will also recognize 21 new IAPR Fellows selected from a field of very strong candidates. In addition, a number of Best Scientific Paper and Best Student Paper awards will be presented, along with the Best Industry Related Paper Award and the Piero Zamperoni Best Student Paper Award. Congratulations to the recipients of these very well-deserved awards!

I would like to close by again thanking everyone involved in making ICPR 2024 a tremendous success; your hard work is deeply appreciated. These thanks extend to all who chaired the various aspects of the conference and the associated workshops, my ExCo colleagues, and the IAPR Standing and Technical Committees. Linda O'Gorman, the IAPR Secretariat, deserves special recognition for her experience, historical perspective, and attention to detail when it comes to supporting many of the IAPR's most important activities. Her tasks became so numerous that she recently got support from Carolyn Buckley (layout, newsletter), Ugur Halici (ICPR matters), and Rosemary Stramka (secretariat). The IAPR website got a completely new design. Ed Sobczak has taken care of our web presence for so many years already. A big thank you to all of you!

This is, of course, the 27th ICPR conference. Knowing that ICPR is organized every two years, and that the first conference in the series (1973!) pre-dated the formal founding of the IAPR by a few years, it is also exciting to consider that we are celebrating over 50 years of ICPR and at the same time approaching the official IAPR 50th anniversary in 2028: you'll get all information you need at ICPR 2024. In the meantime, I offer my thanks and my best wishes to all who are involved in supporting the IAPR throughout the world.

September 2024

Arjan Kuijper
President of the IAPR

Preface

It is our great pleasure to welcome you to the proceedings of the 27th International Conference on Pattern Recognition (ICPR 2024), held in Kolkata, India. The city, formerly known as ‘Calcutta’, is the home of the fabled Indian Statistical Institute (ISI), which has been at the forefront of statistical pattern recognition for almost a century. Concepts like the Mahalanobis distance, Bhattacharyya bound, Cramer–Rao bound, and Fisher–Rao metric were invented by pioneers associated with ISI. The first ICPR (called IJCPR then) was held in 1973, and the second in 1974. Subsequently, ICPR has been held every other year. The International Association for Pattern Recognition (IAPR) was founded in 1978 and became the sponsor of the ICPR series. Over the past 50 years, ICPR has attracted huge numbers of scientists, engineers and students from all over the world and contributed to advancing research, development and applications in pattern recognition technology.

ICPR 2024 was held at the Biswa Bangla Convention Centre, one of the largest such facilities in South Asia, situated just 7 kilometers from Kolkata Airport (CCU). According to ChatGPT “Kolkata is often called the ‘Cultural Capital of India’. The city has a deep connection to literature, music, theater, and art. It was home to Nobel laureate Rabindranath Tagore, and the Bengali film industry has produced globally renowned filmmakers like Satyajit Ray. The city boasts remarkable colonial architecture, with landmarks like Victoria Memorial, Howrah Bridge, and the Indian Museum (the oldest and largest museum in India). Kolkata’s streets are dotted with old mansions and buildings that tell stories of its colonial past. Walking through the city can feel like stepping back into a different era. Finally, Kolkata is also known for its street food.”

ICPR 2024 followed a two-round paper submission format. We received a total of 2135 papers (1501 papers in round-1 submissions, and 634 papers in round-2 submissions). Each paper, on average, received 2.84 reviews, in single-blind mode. For the first-round papers we had a rebuttal option available to authors.

In total, 945 papers (669 from round-1 and 276 from round-2) were accepted for presentation, resulting in an acceptance rate of 44.26%, which is consistent with previous ICPR events. At ICPR 2024 the papers were categorized into six tracks: Artificial Intelligence, Machine Learning for Pattern Analysis; Computer Vision and Robotic Perception; Image, Video, Speech, and Signal Analysis; Biometrics and Human-Machine Interaction; Document and Media Analysis; and Biomedical Image Analysis and Informatics.

The main conference ran over December 2–5, 2024. The main program included the presentation of 188 oral papers (19.89% of the accepted papers), 757 poster papers and 12 competition papers (out of 15 submitted). A total 10 oral sessions were held concurrently in four meeting rooms with a total of 40 oral sessions. In total 24 workshops and 7 tutorials were held on December 1, 2024.

The plenary sessions included three prize lectures and three invited presentations. The prize lectures were delivered by Tin Kam Ho (IBM Research, USA; King Sun

Fu Prize winner), Xiaolong Wang (University of California, San Diego, USA; J.K. Aggarwal Prize winner), and Guoying Zhao (University of Oulu, Finland; Maria Petrou Prize winner). The invited speakers were Timothy Hospedales (University of Edinburgh, UK), Venu Govindaraju (University at Buffalo, USA), and Shuicheng Yan (Skywork AI, Singapore).

Several best paper awards were presented in ICPR: the Piero Zamperoni Award for the best paper authored by a student, the BIRPA Best Industry Related Paper Award, and the Best Paper Awards and Best Student Paper Awards for each of the six tracks of ICPR 2024.

The organization of such a large conference would not be possible without the help of many volunteers. Our special gratitude goes to the Program Chairs (Apostolos Antonacopoulos, Subhasis Chaudhuri, Rama Chellappa and Cheng-Lin Liu), for their leadership in organizing the program. Thanks to our Publication Chairs (Ananda S. Chowdhury and Wataru Ohyama) for handling the overwhelming workload of publishing the conference proceedings. We also thank our Competition Chairs (Richard Zanibbi, Lianwen Jin and Laurence Likforman-Sulem) for arranging 12 important competitions as part of ICPR 2024. We are thankful to our Workshop Chairs (P. Shivakumara, Stephanie Schuckers, Jean-Marc Ogier and Prabir Bhattacharya) and Tutorial Chairs (B.B. Chaudhuri, Michael R. Jenkin and Guoying Zhao) for arranging the workshops and tutorials on emerging topics. ICPR 2024, for the first time, held a Doctoral Consortium. We would like to thank our Doctoral Consortium Chairs (Véronique Eglin, Dan Lopresti and Mayank Vatsa) for organizing it.

Thanks go to the Track Chairs and the meta reviewers who devoted significant time to the review process and preparation of the program. We also sincerely thank the reviewers who provided valuable feedback to the authors.

Finally, we acknowledge the work of other conference committee members, like the Organizing Chairs and Organizing Committee Members, Finance Chairs, Award Chair, Sponsorship Chairs, and Exhibition and Demonstration Chairs, Visa Chair, Publicity Chairs, and Women in ICPR Chairs, whose efforts made this event successful. We also thank our event manager Alpcord Network for their help.

We hope that all the participants found the technical program informative and enjoyed the sights, culture and cuisine of Kolkata.

October 2024

Umapada Pal
Josef Kittler
Anil Jain

Organization

General Chairs

Umapada Pal	Indian Statistical Institute, Kolkata, India
Josef Kittler	University of Surrey, UK
Anil Jain	Michigan State University, USA

Program Chairs

Apostolos Antonacopoulos	University of Salford, UK
Subhasis Chaudhuri	Indian Institute of Technology, Bombay, India
Rama Chellappa	Johns Hopkins University, USA
Cheng-Lin Liu	Institute of Automation, Chinese Academy of Sciences, China

Publication Chairs

Ananda S. Chowdhury	Jadavpur University, India
Wataru Ohyama	Tokyo Denki University, Japan

Competition Chairs

Richard Zanibbi	Rochester Institute of Technology, USA
Lianwen Jin	South China University of Technology, China
Laurence Likforman-Sulem	Télécom Paris, France

Workshop Chairs

P. Shivakumara	University of Salford, UK
Stephanie Schuckers	Clarkson University, USA
Jean-Marc Ogier	Université de la Rochelle, France
Prabir Bhattacharya	Concordia University, Canada

Tutorial Chairs

B. B. Chaudhuri	Indian Statistical Institute, Kolkata, India
Michael R. Jenkin	York University, Canada
Guoying Zhao	University of Oulu, Finland

Doctoral Consortium Chairs

Véronique Eglin	CNRS, France
Daniel P. Lopresti	Lehigh University, USA
Mayank Vatsa	Indian Institute of Technology, Jodhpur, India

Organizing Chairs

Saumik Bhattacharya	Indian Institute of Technology, Kharagpur, India
Palash Ghosal	Sikkim Manipal University, India

Organizing Committee

Santanu Phadikar	West Bengal University of Technology, India
SK Md Obaidullah	Aliah University, India
Sayantari Ghosh	National Institute of Technology Durgapur, India
Himadri Mukherjee	West Bengal State University, India
Nilamadhaba Tripathy	Clarivate Analytics, USA
Chayan Halder	West Bengal State University, India
Shibaprasad Sen	Techno Main Salt Lake, India

Finance Chairs

Kaushik Roy	West Bengal State University, India
Michael Blumenstein	University of Technology Sydney, Australia

Awards Committee Chair

Arpan Pal	Tata Consultancy Services, India

Sponsorship Chairs

P. J. Narayanan	Indian Institute of Technology, Hyderabad, India
Yasushi Yagi	Osaka University, Japan
Venu Govindaraju	University at Buffalo, USA
Alberto Bel Bimbo	Università di Firenze, Italy

Exhibition and Demonstration Chairs

Arjun Jain	FastCode AI, India
Agnimitra Biswas	National Institute of Technology, Silchar, India

International Liaison, Visa Chair

Balasubramanian Raman	Indian Institute of Technology, Roorkee, India

Publicity Chairs

Dipti Prasad Mukherjee	Indian Statistical Institute, Kolkata, India
Bob Fisher	University of Edinburgh, UK
Xiaojun Wu	Jiangnan University, China

Women in ICPR Chairs

Ingela Nystrom	Uppsala University, Sweden
Alexandra B. Albu	University of Victoria, Canada
Jing Dong	Institute of Automation, Chinese Academy of Sciences, China
Sarbani Palit	Indian Statistical Institute, Kolkata, India

Event Manager

Alpcord Network

Track Chairs – Artificial Intelligence, Machine Learning for Pattern Analysis

Larry O’Gorman	Nokia Bell Labs, USA
Dacheng Tao	University of Sydney, Australia
Petia Radeva	University of Barcelona, Spain
Susmita Mitra	Indian Statistical Institute, Kolkata, India
Jiliang Tang	Michigan State University, USA

Track Chairs – Computer and Robot Vision

C. V. Jawahar	International Institute of Information Technology (IIIT), Hyderabad, India
João Paulo Papa	São Paulo State University, Brazil
Maja Pantic	Imperial College London, UK
Gang Hua	Dolby Laboratories, USA
Junwei Han	Northwestern Polytechnical University, China

Track Chairs – Image, Speech, Signal and Video Processing

P. K. Biswas	Indian Institute of Technology, Kharagpur, India
Shang-Hong Lai	National Tsing Hua University, Taiwan
Hugo Jair Escalante	INAOE, CINVESTAV, Mexico
Sergio Escalera	Universitat de Barcelona, Spain
Prem Natarajan	University of Southern California, USA

Track Chairs – Biometrics and Human Computer Interaction

Richa Singh	Indian Institute of Technology, Jodhpur, India
Massimo Tistarelli	University of Sassari, Italy
Vishal Patel	Johns Hopkins University, USA
Wei-Shi Zheng	Sun Yat-sen University, China
Jian Wang	Snap, USA

Track Chairs – Document Analysis and Recognition

Xiang Bai	Huazhong University of Science and Technology, China
David Doermann	University at Buffalo, USA
Josep Llados	Universitat Autònoma de Barcelona, Spain
Mita Nasipuri	Jadavpur University, India

Track Chairs – Biomedical Imaging and Bioinformatics

Jayanta Mukhopadhyay	Indian Institute of Technology, Kharagpur, India
Xiaoyi Jiang	Universität Münster, Germany
Seong-Whan Lee	Korea University, Korea

Metareviewers (Conference Papers and Competition Papers)

Wael Abd-Almageed	University of Southern California, USA
Maya Aghaei	NHL Stenden University, Netherlands
Alireza Alaei	Southern Cross University, Australia
Rajagopalan N. Ambasamudram	Indian Institute of Technology, Madras, India
Suyash P. Awate	Indian Institute of Technology, Bombay, India
Inci M. Baytas	Bogazici University, Turkey
Aparna Bharati	Lehigh University, USA
Brojeshwar Bhowmick	Tata Consultancy Services, India
Jean-Christophe Burie	University of La Rochelle, France
Gustavo Carneiro	University of Surrey, UK
Chee Seng Chan	Universiti Malaya, Malaysia
Sumohana S. Channappayya	Indian Institute of Technology, Hyderabad, India
Dongdong Chen	Microsoft, USA
Shengyong Chen	Tianjin University of Technology, China
Jun Cheng	Institute for Infocomm Research, A*STAR, Singapore
Albert Clapés	University of Barcelona, Spain
Oscar Dalmau	Center for Research in Mathematics, Mexico

Tyler Derr	Vanderbilt University, USA
Abhinav Dhall	Indian Institute of Technology, Ropar, India
Bo Du	Wuhan University, China
Yuxuan Du	University of Sydney, Australia
Ayman S. El-Baz	University of Louisville, USA
Francisco Escolano	University of Alicante, Spain
Siamac Fazli	Nazarbayev University, Kazakhstan
Jianjiang Feng	Tsinghua University, China
Gernot A. Fink	TU Dortmund University, Germany
Alicia Fornes	CVC, Spain
Junbin Gao	University of Sydney, Australia
Yan Gao	Amazon, USA
Yongsheng Gao	Griffith University, Australia
Caren Han	University of Melbourne, Australia
Ran He	Institute of Automation, Chinese Academy of Sciences, China
Tin Kam Ho	IBM, USA
Di Huang	Beihang University, China
Kaizhu Huang	Duke Kunshan University, China
Donato Impedovo	University of Bari, Italy
Julio Jacques	University of Barcelona and Computer Vision Center, Spain
Lianwen Jin	South China University of Technology, China
Wei Jin	Emory University, USA
Danilo Samuel Jodas	São Paulo State University, Brazil
Manjunath V. Joshi	DA-IICT, India
Jayashree Kalpathy-Cramer	Massachusetts General Hospital, USA
Dimosthenis Karatzas	Computer Vision Centre, Spain
Hamid Karimi	Utah State University, USA
Baiying Lei	Shenzhen University, China
Guoqi Li	Chinese Academy of Sciences, and Peng Cheng Lab, China
Laurence Likforman-Sulem	Institut Polytechnique de Paris/Télécom Paris, France
Aishan Liu	Beihang University, China
Bo Liu	Bytedance, USA
Chen Liu	Clarkson University, USA
Cheng-Lin Liu	Institute of Automation, Chinese Academy of Sciences, China
Hongmin Liu	University of Science and Technology Beijing, China
Hui Liu	Michigan State University, USA

Jing Liu	Institute of Automation, Chinese Academy of Sciences, China
Li Liu	University of Oulu, Finland
Qingshan Liu	Nanjing University of Posts and Telecommunications, China
Adrian P. Lopez-Monroy	Centro de Investigacion en Matematicas AC, Mexico
Daniel P. Lopresti	Lehigh University, USA
Shijian Lu	Nanyang Technological University, Singapore
Yong Luo	Wuhan University, China
Andreas K. Maier	FAU Erlangen-Nuremberg, Germany
Davide Maltoni	University of Bologna, Italy
Hong Man	Stevens Institute of Technology, USA
Lingtong Min	Northwestern Polytechnical University, China
Paolo Napoletano	University of Milano-Bicocca, Italy
Kamal Nasrollahi	Milestone Systems, Aalborg University, Denmark
Marcos Ortega	University of A Coruña, Spain
Shivakumara Palaiahnakote	University of Salford, UK
P. Jonathon Phillips	NIST, USA
Filiberto Pla	University Jaume I, Spain
Ajit Rajwade	Indian Institute of Technology, Bombay, India
Shanmuganathan Raman	Indian Institute of Technology, Gandhinagar, India
Imran Razzak	UNSW, Australia
Beatriz Remeseiro	University of Oviedo, Spain
Gustavo Rohde	University of Virginia, USA
Partha Pratim Roy	Indian Institute of Technology, Roorkee, India
Sanjoy K. Saha	Jadavpur University, India
Joan Andreu Sánchez	Universitat Politècnica de València, Spain
Claudio F. Santos	UFSCar, Brazil
Shin'ichi Satoh	National Institute of Informatics, Japan
Stephanie Schuckers	Clarkson University, USA
Srirangaraj Setlur	University at Buffalo, SUNY, USA
Debdoot Sheet	Indian Institute of Technology, Kharagpur, India
Jun Shen	University of Wollongong, Australia
Li Shen	JD Explore Academy, China
Chen Shengyong	Zhejiang University of technology and Tianjin University of Technology, China
Andy Song	RMIT University, Australia
Akihiro Sugimoto	National Institute of Informatics, Japan
Qianru Sun	Singapore Management University, Singapore
Arijit Sur	Indian Institute of Technology, Guwahati, India
Estefania Talavera	University of Twente, Netherlands

Wei Tang	University of Illinois at Chicago, USA
Joao M. Tavares	Universidade do Porto, Portugal
Jun Wan	NLPR, CASIA, China
Le Wang	Xi'an Jiaotong University, China
Lei Wang	Australian National University, Australia
Xiaoyang Wang	Tencent AI Lab, USA
Xinggang Wang	Huazhong University of Science and Technology, China
Xiao-Jun Wu	Jiangnan University, China
Yiding Yang	Bytedance, China
Xiwen Yao	Northwestern Polytechnical University, China
Xu-Cheng Yin	University of Science and Technology Beijing, China
Baosheng Yu	University of Sydney, Australia
Shiqi Yu	Southern University of Science and Technology, China
Xin Yuan	Westlake University, China
Yibing Zhan	JD Explore Academy, China
Jing Zhang	University of Sydney, Australia
Lefei Zhang	Wuhan University, China
Min-Ling Zhang	Southeast University, China
Wenbin Zhang	Florida International University, USA
Jiahuan Zhou	Peking University, China
Sanping Zhou	Xi'an Jiaotong University, China
Tianyi Zhou	University of Maryland, USA
Lei Zhu	Shandong Normal University, China
Pengfei Zhu	Tianjin University, China
Wangmeng Zuo	Harbin Institute of Technology, China

Reviewers (Competition Papers)

Liangcai Gao
Mingxin Huang
Lei Kang
Wenhui Liao
Yuliang Liu
Yongxin Shi
Da-Han Wang
Yang Xue
Wentao Yang
Jiaxin Zhang
Yiwu Zhong

Reviewers (Conference Papers)

Aakanksha Aakanksha
Aayush Singla
Abdul Muqeet
Abhay Yadav
Abhijeet Vijay Nandedkar
Abhimanyu Sahu
Abhinav Rajvanshi
Abhisek Ray
Abhishek Shrivastava
Abhra Chaudhuri
Aditi Roy
Adriano Simonetto
Adrien Maglo
Ahmed Abdulkadir
Ahmed Boudissa
Ahmed Hamdi
Ahmed Rida Sekkat
Ahmed Sharafeldeen
Aiman Farooq
Aishwarya Venkataramanan
Ajay Kumar
Ajay Kumar Reddy Poreddy
Ajita Rattani
Ajoy Mondal
Akbar K.
Akbar Telikani
Akshay Agarwal
Akshit Jindal
Al Zadid Sultan Bin Habib
Albert Clapés
Alceu Britto
Alejandro Peña
Alessandro Ortis
Alessia Auriemma Citarella
Alexandre Stenger
Alexandros Sopasakis
Alexia Toumpa
Ali Khan
Alik Pramanick
Alireza Alaei
Alper Yilmaz
Aman Verma
Amit Bhardwaj
Amit More
Amit Nandedkar
Amitava Chatterjee
Amos L. Abbott
Amrita Mohan
Anand Mishra
Ananda S. Chowdhury
Anastasia Zakharova
Anastasios L. Kesidis
Andras Horvath
Andre Gustavo Hochuli
André P. Kelm
Andre Wyzykowski
Andrea Bottino
Andrea Lagorio
Andrea Torsello
Andreas Fischer
Andreas K. Maier
Andreu Girbau Xalabarder
Andrew Beng Jin Teoh
Andrew Shin
Andy J. Ma
Aneesh S. Chivukula
Ángela Casado-García
Anh Quoc Nguyen
Anindya Sen
Anirban Saha
Anjali Gautam
Ankan Bhattacharyya
Ankit Jha
Anna Scius-Bertrand
Annalisa Franco
Antoine Doucet
Antonino Staiano
Antonio Fernández
Antonio Parziale
Anu Singha
Anustup Choudhury
Anwesan Pal
Anwesha Sengupta
Archisman Adhikary
Arjan Kuijper
Arnab Kumar Das

Arnav Bhavsar
Arnav Varma
Arpita Dutta
Arshad Jamal
Artur Jordao
Arunkumar Chinnaswamy
Aryan Jadon
Aryaz Baradarani
Ashima Anand
Ashis Dhara
Ashish Phophalia
Ashok K. Bhateja
Ashutosh Vaish
Ashwani Kumar
Asifuzzaman Lasker
Atefeh Khoshkhahtinat
Athira Nambiar
Attilio Fiandrotti
Avandra S. Hemachandra
Avik Hati
Avinash Sharma
B. H. Shekar
B. Uma Shankar
Bala Krishna Thunakala
Balaji Tk
Balázs Pálffy
Banafsheh Adami
Bang-Dang Pham
Baochang Zhang
Baodi Liu
Bashirul Azam Biswas
Beiduo Chen
Benedikt Kottler
Beomseok Oh
Berkay Aydin
Berlin S. Shaheema
Bertrand Kerautret
Bettina Finzel
Bhavana Singh
Bibhas C. Dhara
Bilge Gunsel
Bin Chen
Bin Li
Bin Liu
Bin Yao
Bin-Bin Jia
Binbin Yong
Bindita Chaudhuri
Bindu Madhavi Tummala
Binh M. Le
Bi-Ru Dai
Bo Huang
Bo Jiang
Bob Zhang
Bowen Liu
Bowen Zhang
Boyang Zhang
Boyu Diao
Boyun Li
Brian M. Sadler
Bruce A. Maxwell
Bryan Bo Cao
Buddhika L. Semage
Bushra Jalil
Byeong-Seok Shin
Byung-Gyu Kim
Caihua Liu
Cairong Zhao
Camille Kurtz
Carlos A. Caetano
Carlos D. Martã-Nez-Hinarejos
Ce Wang
Cevahir Cigla
Chakravarthy Bhagvati
Chandrakanth Vipparla
Changchun Zhang
Changde Du
Changkun Ye
Changxu Cheng
Chao Fan
Chao Guo
Chao Qu
Chao Wen
Chayan Halder
Che-Jui Chang
Chen Feng
Chenan Wang
Cheng Yu
Chenghao Qian
Cheng-Lin Liu

Chengxu Liu
Chenru Jiang
Chensheng Peng
Chetan Ralekar
Chih-Wei Lin
Chih-Yi Chiu
Chinmay Sahu
Chintan Patel
Chintan Shah
Chiranjoy Chattopadhyay
Chong Wang
Choudhary Shyam Prakash
Christophe Charrier
Christos Smailis
Chuanwei Zhou
Chun-Ming Tsai
Chunpeng Wang
Ciro Russo
Claudio De Stefano
Claudio F. Santos
Claudio Marrocco
Connor Levenson
Constantine Dovrolis
Constantine Kotropoulos
Dai Shi
Dakshina Ranjan Kisku
Dan Anitei
Dandan Zhu
Daniela Pamplona
Danli Wang
Danqing Huang
Daoan Zhang
Daqing Hou
David A. Clausi
David Freire Obregon
David Münch
David Pujol Perich
Davide Marelli
De Zhang
Debalina Barik
Debapriya Roy (Kundu)
Debashis Das
Debashis Das Chakladar
Debi Prosad Dogra
Debraj D. Basu
Decheng Liu
Deen Dayal Mohan
Deep A. Patel
Deepak Kumar
Dengpan Liu
Denis Coquenet
Désiré Sidibé
Devesh Walawalkar
Dewan Md. Farid
Di Ming
Di Qiu
Di Yuan
Dian Jia
Dianmo Sheng
Diego Thomas
Diganta Saha
Dimitri Bulatov
Dimpy Varshni
Dingcheng Yang
Dipanjan Das
Dipanjyoti Paul
Divya Biligere Shivanna
Divya Saxena
Divya Sharma
Dmitrii Matveichev
Dmitry Minskiy
Dmitry V. Sorokin
Dong Zhang
Donghua Wang
Donglin Zhang
Dongming Wu
Dongqiangzi Ye
Dongqing Zou
Dongrui Liu
Dongyang Zhang
Dongzhan Zhou
Douglas Rodrigues
Duarte Folgado
Duc Minh Vo
Duoxuan Pei
Durai Arun Pannir Selvam
Durga Bhavani S.
Eckart Michaelsen
Elena Goyanes
Élodie Puybareau

Emanuele Vivoli
Emna Ghorbel
Enrique Naredo
Enyu Cai
Eric Patterson
Ernest Valveny
Eva Blanco-Mallo
Eva Breznik
Evangelos Sartinas
Fabio Solari
Fabiola De Marco
Fan Wang
Fangda Li
Fangyuan Lei
Fangzhou Lin
Fangzhou Luo
Fares Bougourzi
Farman Ali
Fatiha Mokdad
Fei Shen
Fei Teng
Fei Zhu
Feiyan Hu
Felipe Gomes Oliveira
Feng Li
Fengbei Liu
Fenghua Zhu
Fillipe D. M. De Souza
Flavio Piccoli
Flavio Prieto
Florian Kleber
Francesc Serratosa
Francesco Bianconi
Francesco Castro
Francesco Ponzio
Francisco Javier Hernández López
Frédéric Rayar
Furkan Osman Kar
Fushuo Huo
Fuxiao Liu
Fu-Zhao Ou
Gabriel Turinici
Gabrielle Flood
Gajjala Viswanatha Reddy
Gaku Nakano
Galal Binamakhashen
Ganesh Krishnasamy
Gang Pan
Gangyan Zeng
Gani Rahmon
Gaurav Harit
Gennaro Vessio
Genoveffa Tortora
George Azzopardi
Gerard Ortega
Gerardo E. Altamirano-Gomez
Gernot A. Fink
Gibran Benitez-Garcia
Gil Ben-Artzi
Gilbert Lim
Giorgia Minello
Giorgio Fumera
Giovanna Castellano
Giovanni Puglisi
Giulia Orrù
Giuliana Ramella
Gökçe Uludoğan
Gopi Ramena
Gorthi Rama Krishna Sai Subrahmanyam
Gourav Datta
Gowri Srinivasa
Gozde Sahin
Gregory Randall
Guanjie Huang
Guanjun Li
Guanwen Zhang
Guanyu Xu
Guanyu Yang
Guanzhou Ke
Guhnoo Yun
Guido Borghi
Guilherme Brandão Martins
Guillaume Caron
Guillaume Tochon
Guocai Du
Guohao Li
Guoqiang Zhong
Guorong Li
Guotao Li
Gurman Gill

Haechang Lee
Haichao Zhang
Haidong Xie
Haifeng Zhao
Haimei Zhao
Hainan Cui
Haixia Wang
Haiyan Guo
Hakime Ozturk
Hamid Kazemi
Han Gao
Hang Zou
Hanjia Lyu
Hanjoo Cho
Hanqing Zhao
Hanyuan Liu
Hanzhou Wu
Hao Li
Hao Meng
Hao Sun
Hao Wang
Hao Xing
Hao Zhao
Haoan Feng
Haodi Feng
Haofeng Li
Haoji Hu
Haojie Hao
Haojun Ai
Haopeng Zhang
Haoran Li
Haoran Wang
Haorui Ji
Haoxiang Ma
Haoyu Chen
Haoyue Shi
Harald Koestler
Harbinder Singh
Harris V. Georgiou
Hasan F. Ates
Hasan S. M. Al-Khaffaf
Hatef Otroshi Shahreza
Hebeizi Li
Heng Zhang
Hengli Wang
Hengyue Liu
Hertog Nugroho
Hieyong Jeong
Himadri Mukherjee
Hoai Ngo
Hoda Mohaghegh
Hong Liu
Hong Man
Hongcheng Wang
Hongjian Zhan
Hongxi Wei
Hongyu Hu
Hoseong Kim
Hossein Ebrahimnezhad
Hossein Malekmohamadi
Hrishav Bakul Barua
Hsueh-Yi Sean Lin
Hua Wei
Huafeng Li
Huali Xu
Huaming Chen
Huan Wang
Huang Chen
Huanran Chen
Hua-Wen Chang
Huawen Liu
Huayi Zhan
Hugo Jair Escalante
Hui Chen
Hui Li
Huichen Yang
Huiqiang Jiang
Huiyuan Yang
Huizi Yu
Hung T. Nguyen
Hyeongyu Kim
Hyeonjeong Park
Hyeonjun Lee
Hymalai Bello
Hyung-Gun Chi
Hyunsoo Kim
I-Chen Lin
Ik Hyun Lee
Ilan Shimshoni
Imad Eddine Toubal

Imran Sarker
Inderjot Singh Saggu
Indrani Mukherjee
Indranil Sur
Ines Rieger
Ioannis Pierros
Irina Rabaev
Ivan V. Medri
J. Rafid Siddiqui
Jacek Komorowski
Jacopo Bonato
Jacson Rodrigues Correia-Silva
Jaekoo Lee
Jaime Cardoso
Jakob Gawlikowski
Jakub Nalepa
James L. Wayman
Jan Čech
Jangho Lee
Jani Boutellier
Javier Gurrola-Ramos
Javier Lorenzo-Navarro
Jayasree Saha
Jean Lee
Jean Paul Barddal
Jean-Bernard Hayet
Jean-Philippe G. Tarel
Jean-Yves Ramel
Jenny Benois-Pineau
Jens Bayer
Jerin Geo James
Jesús Miguel García-Gorrostieta
Jia Qu
Jiahong Chen
Jiaji Wang
Jian Hou
Jian Liang
Jian Xu
Jian Zhu
Jianfeng Lu
Jianfeng Ren
Jiangfan Liu
Jianguo Wang
Jiangyan Yi
Jiangyong Duan
Jianhua Yang
Jianhua Zhang
Jianhui Chen
Jianjia Wang
Jianli Xiao
Jianqiang Xiao
Jianwu Wang
Jianxin Zhang
Jianxiong Gao
Jianxiong Zhou
Jianyu Wang
Jianzhong Wang
Jiaru Zhang
Jiashu Liao
Jiaxin Chen
Jiaxin Lu
Jiaxing Ye
Jiaxuan Chen
Jiaxuan Li
Jiayi He
Jiayin Lin
Jie Ou
Jiehua Zhang
Jiejie Zhao
Jignesh S. Bhatt
Jin Gao
Jin Hou
Jin Hu
Jin Shang
Jing Tian
Jing Yu Chen
Jingfeng Yao
Jinglun Feng
Jingtong Yue
Jingwei Guo
Jingwen Xu
Jingyuan Xia
Jingzhe Ma
Jinhong Wang
Jinjia Wang
Jinlai Zhang
Jinlong Fan
Jinming Su
Jinrong He
Jintao Huang

Jinwoo Ahn
Jinwoo Choi
Jinyang Liu
Jinyu Tian
Jionghao Lin
Jiuding Duan
Jiwei Shen
Jiyan Pan
Jiyoun Kim
João Papa
Johan Debayle
John Atanbori
John Wilson
John Zhang
Jónathan Heras
Joohi Chauhan
Jorge Calvo-Zaragoza
Jorge Figueroa
Jorma Laaksonen
José Joaquim De Moura Ramos
Jose Vicent
Joseph Damilola Akinyemi
Josiane Zerubia
Juan Wen
Judit Szücs
Juepeng Zheng
Juha Roning
Jumana H. Alsubhi
Jun Cheng
Jun Ni
Jun Wan
Junghyun Cho
Junjie Liang
Junjie Ye
Junlin Hu
Juntong Ni
Junxin Lu
Junxuan Li
Junyaup Kim
Junyeong Kim
Jürgen Seiler
Jushang Qiu
Juyang Weng
Jyostna Devi Bodapati
Jyoti Singh Kirar

Kai Jiang
Kaiqiang Song
Kalidas Yeturu
Kalle Åström
Kamalakar Vijay Thakare
Kang Gu
Kang Ma
Kanji Tanaka
Karthik Seemakurthy
Kaushik Roy
Kavisha Jayathunge
Kazuki Uehara
Ke Shi
Keigo Kimura
Keiji Yanai
Kelton A. P. Costa
Kenneth Camilleri
Kenny Davila
Ketan Atul Bapat
Ketan Kotwal
Kevin Desai
Keyu Long
Khadiga Mohamed Ali
Khakon Das
Khan Muhammad
Kilho Son
Kim-Ngan Nguyen
Kishan Kc
Kishor P. Upla
Klaas Dijkstra
Komal Bharti
Konstantinos Triaridis
Kostas Ioannidis
Koyel Ghosh
Kripabandhu Ghosh
Krishnendu Ghosh
Kshitij S. Jadhav
Kuan Yan
Kun Ding
Kun Xia
Kun Zeng
Kunal Banerjee
Kunal Biswas
Kunchi Li
Kurban Ubul

Lahiru N. Wijayasingha
Laines Schmalwasser
Lakshman Mahto
Lala Shakti Swarup Ray
Lale Akarun
Lan Yan
Lawrence Amadi
Lee Kang Il
Lei Fan
Lei Shi
Lei Wang
Leonardo Rossi
Lequan Lin
Levente Tamas
Li Bing
Li Li
Li Ma
Li Song
Lia Morra
Liang Xie
Liang Zhao
Lianwen Jin
Libing Zeng
Lidia Sánchez-González
Lidong Zeng
Lijun Li
Likang Wang
Lili Zhao
Lin Chen
Lin Huang
Linfei Wang
Ling Lo
Lingchen Meng
Lingheng Meng
Lingxiao Li
Lingzhong Fan
Liqi Yan
Liqiang Jing
Lisa Gutzeit
Liu Ziyi
Liushuai Shi
Liviu-Daniel Stefan
Liyuan Ma
Liyun Zhu
Lizuo Jin

Longteng Guo
Lorena Álvarez Rodríguez
Lorenzo Putzu
Lu Leng
Lu Pang
Lu Wang
Luan Pham
Luc Brun
Luca Guarnera
Luca Piano
Lucas Alexandre Ramos
Lucas Goncalves
Lucas M. Gago
Luigi Celona
Luis C. S. Afonso
Luis Gerardo De La Fraga
Luis S. Luevano
Luis Teixeira
Lunke Fei
M. Hassaballah
Maddimsetti Srinivas
Mahendran N.
Mahesh Mohan M. R.
Maiko Lie
Mainak Singha
Makoto Hirose
Malay Bhattacharyya
Mamadou Dian Bah
Man Yao
Manali J. Patel
Manav Prabhakar
Manikandan V. M.
Manish Bhatt
Manjunath Shantharamu
Manuel Curado
Manuel Günther
Manuel Marques
Marc A. Kastner
Marc Chaumont
Marc Cheong
Marc Lalonde
Marco Cotogni
Marcos C. Santana
Mario Molinara
Mariofanna Milanova

Markus Bauer
Marlon Becker
Mårten Wadenbäck
Martin G. Ljungqvist
Martin Kampel
Martina Pastorino
Marwan Torki
Masashi Nishiyama
Masayuki Tanaka
Massimo O. Spata
Matteo Ferrara
Matthew D. Dawkins
Matthew Gadd
Matthew S. Watson
Maura Pintor
Max Ehrlich
Maxim Popov
Mayukh Das
Md Baharul Islam
Md Sajid
Meghna Kapoor
Meghna P. Ayyar
Mei Wang
Meiqi Wu
Melissa L. Tijink
Meng Li
Meng Liu
Meng-Luen Wu
Mengnan Liu
Mengxi China Guo
Mengya Han
Michaël Clément
Michal Kawulok
Mickael Coustaty
Miguel Domingo
Milind G. Padalkar
Ming Liu
Ming Ma
Mingchen Feng
Mingde Yao
Minghao Li
Mingjie Sun
Ming-Kuang Daniel Wu
Mingle Xu
Mingyong Li
Mingyuan Jiu
Minh P. Nguyen
Minh Q. Tran
Minheng Ni
Minsu Kim
Minyi Zhao
Mirko Paolo Barbato
Mo Zhou
Modesto Castrillón-Santana
Mohamed Amine Mezghich
Mohamed Dahmane
Mohamed Elsharkawy
Mohamed Yousuf
Mohammad Hashemi
Mohammad Khalooei
Mohammad Khateri
Mohammad Mahdi Dehshibi
Mohammad Sadil Khan
Mohammed Mahmoud
Moises Diaz
Monalisha Mahapatra
Monidipa Das
Mostafa Kamali Tabrizi
Mridul Ghosh
Mrinal Kanti Bhowmik
Muchao Ye
Mugalodi Ramesha Rakesh
Muhammad Rameez Ur Rahman
Muhammad Suhaib Kanroo
Muming Zhao
Munender Varshney
Munsif Ali
Na Lv
Nader Karimi
Nagabhushan Somraj
Nakkwan Choi
Nakul Agarwal
Nan Pu
Nan Zhou
Nancy Mehta
Nand Kumar Yadav
Nandakishor Nandakishor
Nandyala Hemachandra
Nanfeng Jiang
Narayan Hegde

Narayan Ji Mishra
Narayan Vetrekar
Narendra D. Londhe
Nathalie Girard
Nati Ofir
Naval Kishore Mehta
Nazmul Shahadat
Neeti Narayan
Neha Bhargava
Nemanja Djuric
Newlin Shebiah R.
Ngo Ba Hung
Nhat-Tan Bui
Niaz Ahmad
Nick Theisen
Nicolas Passat
Nicolas Ragot
Nicolas Sidere
Nikolaos Mitianoudis
Nikolas Ebert
Nilah Ravi Nair
Nilesh A. Ahuja
Nilkanta Sahu
Nils Murrugarra-Llerena
Nina S. T. Hirata
Ninad Aithal
Ning Xu
Ningzhi Wang
Niraj Kumar
Nirmal S. Punjabi
Nisha Varghese
Norio Tagawa
Obaidullah Md Sk
Oguzhan Ulucan
Olfa Mechi
Oliver Tüselmann
Orazio Pontorno
Oriol Ramos Terrades
Osman Akin
Ouadi Beya
Ozge Mercanoglu Sincan
Pabitra Mitra
Padmanabha Reddy Y. C. A.
Palaash Agrawal
Palaiahnakote Shivakumara
Palash Ghosal
Pallav Dutta
Paolo Rota
Paramanand Chandramouli
Paria Mehrani
Parth Agrawal
Partha Basuchowdhuri
Patrick Horain
Pavan Kumar
Pavan Kumar Anasosalu Vasu
Pedro Castro
Peipei Li
Peipei Yang
Peisong Shen
Peiyu Li
Peng Li
Pengfei He
Pengrui Quan
Pengxin Zeng
Pengyu Yan
Peter Eisert
Petra Gomez-Krämer
Pierrick Bruneau
Ping Cao
Pingping Zhang
Pintu Kumar
Pooja Kumari
Pooja Sahani
Prabhu Prasad Dev
Pradeep Kumar
Pradeep Singh
Pranjal Sahu
Prasun Roy
Prateek Keserwani
Prateek Mittal
Praveen Kumar Chandaliya
Praveen Tirupattur
Pravin Nair
Preeti Gopal
Preety Singh
Prem Shanker Yadav
Prerana Mukherjee
Prerna A. Mishra
Prianka Dey
Priyanka Mudgal

Qc Kha Ng
Qi Li
Qi Ming
Qi Wang
Qi Zuo
Qian Li
Qiang Gan
Qiang He
Qiang Wu
Qiangqiang Zhou
Qianli Zhao
Qiansen Hong
Qiao Wang
Qidong Huang
Qihua Dong
Qin Yuke
Qing Guo
Qingbei Guo
Qingchao Zhang
Qingjie Liu
Qinhong Yang
Qiushi Shi
Qixiang Chen
Quan Gan
Quanlong Guan
Rachit Chhaya
Radu Tudor Ionescu
Rafal Zdunek
Raghavendra Ramachandra
Rahimul I. Mazumdar
Rahul Kumar Ray
Rajib Dutta
Rajib Ghosh
Rakesh Kumar
Rakesh Paul
Rama Chellappa
Rami O. Skaik
Ramon Aranda
Ran Wei
Ranga Raju Vatsavai
Ranganath Krishnan
Rasha Friji
Rashmi S.
Razaib Tariq
Rémi Giraud
René Schuster
Renlong Hang
Renrong Shao
Renu Sharma
Reza Sadeghian
Richard Zanibbi
Rimon Elias
Rishabh Shukla
Rita Delussu
Riya Verma
Robert J. Ravier
Robert Sablatnig
Robin Strand
Rocco Pietrini
Rocio Diaz Martin
Rocio Gonzalez-Diaz
Rohit Venkata Sai Dulam
Romain Giot
Romi Banerjee
Ru Wang
Ruben Machucho
Ruddy Théodose
Ruggero Pintus
Rui Deng
Rui P. Paiva
Rui Zhao
Ruifan Li
Ruigang Fu
Ruikun Li
Ruirui Li
Ruixiang Jiang
Ruowei Jiang
Rushi Lan
Rustam Zhumagambetov
S. Amutha
S. Divakar Bhat
Sagar Goyal
Sahar Siddiqui
Sahbi Bahroun
Sai Karthikeya Vemuri
Saibal Dutta
Saihui Hou
Sajad Ahmad Rather
Saksham Aggarwal
Sakthi U.

Salimeh Sekeh
Samar Bouazizi
Samia Boukir
Samir F. Harb
Samit Biswas
Samrat Mukhopadhyay
Samriddha Sanyal
Sandika Biswas
Sandip Purnapatra
Sanghyun Jo
Sangwoo Cho
Sanjay Kumar
Sankaran Iyer
Sanket Biswas
Santanu Roy
Santosh D. Pandure
Santosh Ku Behera
Santosh Nanabhau Palaskar
Santosh Prakash Chouhan
Sarah S. Alotaibi
Sasanka Katreddi
Sathyanarayanan N. Aakur
Saurabh Yadav
Sayan Rakshit
Scott McCloskey
Sebastian Bunda
Sejuti Rahman
Selim Aksoy
Sen Wang
Seraj A. Mostafa
Shanmuganathan Raman
Shao-Yuan Lo
Shaoyuan Xu
Sharia Arfin Tanim
Shehreen Azad
Sheng Wan
Shengdong Zhang
Shengwei Qin
Shenyuan Gao
Sherry X. Chen
Shibaprasad Sen
Shigeaki Namiki
Shiguang Liu
Shijie Ma
Shikun Li
Shinichiro Omachi
Shirley David
Shishir Shah
Shiv Ram Dubey
Shiva Baghel
Shivanand S. Gornale
Shogo Sato
Shotaro Miwa
Shreya Ghosh
Shreya Goyal
Shuai Su
Shuai Wang
Shuai Zheng
Shuaifeng Zhi
Shuang Qiu
Shuhei Tarashima
Shujing Lyu
Shuliang Wang
Shun Zhang
Shunming Li
Shunxin Wang
Shuping Zhao
Shuquan Ye
Shuwei Huo
Shuyue Lan
Shyi-Chyi Cheng
Si Chen
Siddarth Ravichandran
Sihan Chen
Siladittya Manna
Silambarasan Elkana Ebinazer
Simon Benaïchouche
Simon S. Woo
Simone Caldarella
Simone Milani
Simone Zini
Sina Lotfian
Sitao Luan
Sivaselvan B.
Siwei Li
Siwei Wang
Siwen Luo
Siyu Chen
Sk Aziz Ali
Sk Md Obaidullah

Sneha Shukla
Snehasis Banerjee
Snehasis Mukherjee
Snigdha Sen
Sofia Casarin
Soheila Farokhi
Soma Bandyopadhyay
Son Minh Nguyen
Son Xuan Ha
Sonal Kumar
Sonam Gupta
Sonam Nahar
Song Ouyang
Sotiris Kotsiantis
Souhaila Djaffal
Soumen Biswas
Soumen Sinha
Soumitri Chattopadhyay
Souvik Sengupta
Spiros Kostopoulos
Sreeraj Ramachandran
Sreya Banerjee
Srikanta Pal
Srinivas Arukonda
Stephane A. Guinard
Su O. Ruan
Subhadip Basu
Subhajit Paul
Subhankar Ghosh
Subhankar Mishra
Subhankar Roy
Subhash Chandra Pal
Subhayu Ghosh
Sudip Das
Sudipta Banerjee
Suhas Pillai
Sujit Das
Sukalpa Chanda
Sukhendu Das
Suklav Ghosh
Suman K. Ghosh
Suman Samui
Sumit Mishra
Sungho Suh
Sunny Gupta
Suraj Kumar Pandey
Surendrabikram Thapa
Suresh Sundaram
Sushil Bhattacharjee
Susmita Ghosh
Swakkhar Shatabda
Syed Ms Islam
Syed Tousiful Haque
Taegyeong Lee
Taihui Li
Takashi Shibata
Takeshi Oishi
Talha Ahmad Siddiqui
Tanguy Gernot
Tangwen Qian
Tanima Bhowmik
Tanpia Tasnim
Tao Dai
Tao Hu
Tao Sun
Taoran Yi
Tapan Shah
Taveena Lotey
Teng Huang
Tengqi Ye
Teresa Alarcon
Tetsuji Ogawa
Thanh Phuong Nguyen
Thanh Tuan Nguyen
Thattapon Surasak
Thibault Napolãon
Thierry Bouwmans
Thinh Truong Huynh Nguyen
Thomas De Min
Thomas E. K. Zielke
Thomas Swearingen
Tianatahina Jimmy Francky Randrianasoa
Tianheng Cheng
Tianjiao He
Tianyi Wei
Tianyuan Zhang
Tianyue Zheng
Tiecheng Song
Tilottama Goswami
Tim Büchner

Tim H. Langer
Tim Raven
Tingkai Liu
Tingting Yao
Tobias Meisen
Toby P. Breckon
Tong Chen
Tonghua Su
Tran Tuan Anh
Tri-Cong Pham
Trishna Saikia
Trung Quang Truong
Tuan T. Nguyen
Tuan Vo Van
Tushar Shinde
Ujjwal Karn
Ukrit Watchareeruetai
Uma Mudenagudi
Umarani Jayaraman
V. S. Malemath
Vallidevi Krishnamurthy
Ved Prakash
Venkata Krishna Kishore Kolli
Venkata R. Vavilthota
Venkatesh Thirugnana Sambandham
Verónica Maria Vasconcelos
Véronique Ve Eglin
Víctor E. Alonso-Pérez
Vinay Palakkode
Vinayak S. Nageli
Vincent J. Whannou De Dravo
Vincenzo Conti
Vincenzo Gattulli
Vineet Padmanabhan
Vishakha Pareek
Viswanath Gopalakrishnan
Vivek Singh Baghel
Vivekraj K.
Vladimir V. Arlazarov
Vu-Hoang Tran
W. Sylvia Lilly Jebarani
Wachirawit Ponghiran
Wafa Khlif
Wang An-Zhi
Wanli Xue
Wataru Ohyama
Wee Kheng Leow
Wei Chen
Wei Cheng
Wei Hua
Wei Lu
Wei Pan
Wei Tian
Wei Wang
Wei Wei
Wei Zhou
Weidi Liu
Weidong Yang
Weijun Tan
Weimin Lyu
Weinan Guan
Weining Wang
Weiqiang Wang
Weiwei Guo
Weixia Zhang
Wei-Xuan Bao
Weizhong Jiang
Wen Xie
Wenbin Qian
Wenbin Tian
Wenbin Wang
Wenbo Zheng
Wenhan Luo
Wenhao Wang
Wen-Hung Liao
Wenjie Li
Wenkui Yang
Wenwen Si
Wenwen Yu
Wenwen Zhang
Wenwu Yang
Wenxi Li
Wenxi Yue
Wenxue Cui
Wenzhuo Liu
Widhiyo Sudiyono
Willem Dijkstra
Wolfgang Fuhl
Xi Zhang
Xia Yuan

Xianda Zhang
Xiang Zhang
Xiangdong Su
Xiang-Ru Yu
Xiangtai Li
Xiangyu Xu
Xiao Guo
Xiao Hu
Xiao Wu
Xiao Yang
Xiaofeng Zhang
Xiaogang Du
Xiaoguang Zhao
Xiaoheng Jiang
Xiaohong Zhang
Xiaohua Huang
Xiaohua Li
Xiao-Hui Li
Xiaolong Sun
Xiaosong Li
Xiaotian Li
Xiaoting Wu
Xiaotong Luo
Xiaoyan Li
Xiaoyang Kang
Xiaoyi Dong
Xin Guo
Xin Lin
Xin Ma
Xinchi Zhou
Xingguang Zhang
Xingjian Leng
Xingpeng Zhang
Xingzheng Lyu
Xinjian Huang
Xinqi Fan
Xinqi Liu
Xinqiao Zhang
Xinrui Cui
Xizhan Gao
Xu Cao
Xu Ouyang
Xu Zhao
Xuan Shen
Xuan Zhou
Xuchen Li
Xuejing Lei
Xuelu Feng
Xueting Liu
Xuewei Li
Xueyi X. Wang
Xugong Qin
Xu-Qian Fan
Xuxu Liu
Xu-Yao Zhang
Yan Huang
Yan Li
Yan Wang
Yan Xia
Yan Zhuang
Yanan Li
Yanan Zhang
Yang Hou
Yang Jiao
Yang Liping
Yang Liu
Yang Qian
Yang Yang
Yang Zhao
Yangbin Chen
Yangfan Zhou
Yanhui Guo
Yanjia Huang
Yanjun Zhu
Yanming Zhang
Yanqing Shen
Yaoming Cai
Yaoxin Zhuo
Yaoyan Zheng
Yaping Zhang
Yaqian Liang
Yarong Feng
Yasmina Benmabrouk
Yasufumi Sakai
Yasutomo Kawanishi
Yazeed Alzahrani
Ye Du
Ye Duan
Yechao Zhang
Yeong-Jun Cho

Yi Huo
Yi Shi
Yi Yu
Yi Zhang
Yibo Liu
Yibo Wang
Yi-Chieh Wu
Yifan Chen
Yifei Huang
Yihao Ding
Yijie Tang
Yikun Bai
Yimin Wen
Yinan Yang
Yin-Dong Zheng
Yinfeng Yu
Ying Dai
Yingbo Li
Yiqiao Li
Yiqing Huang
Yisheng Lv
Yisong Xiao
Yite Wang
Yizhe Li
Yong Wang
Yonghao Dong
Yong-Hyuk Moon
Yongjie Li
Yongqian Li
Yongqiang Mao
Yongxu Liu
Yongyu Wang
Yongzhi Li
Youngha Hwang
Yousri Kessentini
Yu Wang
Yu Zhou
Yuan Tian
Yuan Zhang
Yuanbo Wen
Yuanxin Wang
Yubin Hu
Yubo Huang
Yuchen Ren
Yucheng Xing
Yuchong Yao
Yuecong Min
Yuewei Yang
Yufei Zhang
Yufeng Yin
Yugen Yi
Yuhang Ming
Yujia Zhang
Yujun Ma
Yukiko Kenmochi
Yun Hoyeoung
Yun Liu
Yunhe Feng
Yunxiao Shi
Yuru Wang
Yushun Tang
Yusuf Osmanlioglu
Yusuke Fujita
Yuta Nakashima
Yuwei Yang
Yuwu Lu
Yuxi Liu
Yuya Obinata
Yuyao Yan
Yuzhi Guo
Zaipeng Xie
Zander W. Blasingame
Zedong Wang
Zeliang Zhang
Zexin Ji
Zhanxiang Feng
Zhaofei Yu
Zhe Chen
Zhe Cui
Zhe Liu
Zhe Wang
Zhekun Luo
Zhen Yang
Zhenbo Li
Zhenchun Lei
Zhenfei Zhang
Zheng Liu
Zheng Wang
Zhengming Yu
Zhengyin Du

Zhengyun Cheng
Zhenshen Qu
Zhenwei Shi
Zhenzhong Kuang
Zhi Cai
Zhi Chen
Zhibo Chu
Zhicun Yin
Zhida Huang
Zhida Zhang
Zhifan Gao
Zhihang Ren
Zhihang Yuan
Zhihao Wang
Zhihua Xie
Zhihui Wang
Zhikang Zhang
Zhiming Zou
Zhiqi Shao
Zhiwei Dong
Zhiwei Qi
Zhixiang Wang
Zhixuan Li
Zhiyu Jiang
Zhiyuan Yan
Zhiyuan Yu
Zhiyuan Zhang
Zhong Chen
Zhongwei Teng
Zhongzhan Huang
Zhongzhi Yu
Zhuan Han
Zhuangzhuang Chen
Zhuo Liu
Zhuo Su
Zhuojun Zou
Zhuoyue Wang
Ziang Song
Zicheng Zhang
Zied Mnasri
Zifan Chen
Žiga Babnik
Zijing Chen
Zikai Zhang
Ziling Huang
Zilong Du
Ziqi Cai
Ziqi Zhou
Zi-Rui Wang
Zirui Zhou
Ziwen He
Ziyao Zeng
Ziyi Zhang
Ziyue Xiang
Zonglei Jing
Zongyi Xu

Contents – Part XIX

Geometric Deep Learning in Industrial Scenes: A Large-Scale 3D Synthetic Dataset

Igor P. Maurell[1](✉), Pedro L. Corçaque[1], Cris L. Froes[1], João Francisco S. S. Lemos[1], Felipe G. Oliveira[1,2], and Paulo L. J. Drews-Jr[1]

[1] Computer Sciences Center (C3), Universidade Federal do Rio Grande (FURG), Rio Grande, Rio Grande do Sul, Brazil
{igormaurell,pedrollcorc,crislimafroes,jfsslemos,paulodrews}@furg.br
[2] Institute of Exact Sciences and Technology (ICET), Universidade Federal do Amazonas (UFAM), Itacoatiara, Amazonas, Brazil
felipeoliveira@ufam.edu.br

Abstract. We introduce the LS3DS dataset, a novel collection of large 3D CAD models, meshes, and point clouds of industrial sites. Additionally, the proposed dataset provides a processing pipeline to generate synthetic meshes and point clouds from the CAD models. The open-source pipeline makes the proposed dataset easily applicable to different scenarios, such as the construction industry. The ground truth values can be effortlessly generated to tackle problems such as geometric primitive fitting, a meaningful challenge strongly related to segmentation and shape fitting problems. Furthermore, we present a benchmark addressing the problem of large-scale point cloud geometric primitive fitting. We adapted state-of-the-art deep learning-based methods for the benchmark to process large-scale point clouds. We compared them to a baseline classical approach, which shows challenges in complex, large-scale industrial environments defined by dense and varied geometric distributions. Our paper demonstrates the meaningful contribution of the proposed dataset with a case study presented in the benchmark, opening opportunities for dealing with relevant problems of 3D geometric understanding using learning approaches. The generation pipeline, LS3DS dataset and the weights of the models trained in the benchmark are openly available to use (LS3DS Repository: https://github.com/igormaurell/LS3DS).

Keywords: Large-Scale 3D Dataset · Geometric Deep Learning · Industrial Environments · Point Clouds

1 Introduction

Over the past few decades, the importance of data has increased, driven in part by the growing application of supervised machine learning methods for 3D data [26]. Therefore, a significant challenge in supervised learning is the need for

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 1–19, 2025.
https://doi.org/10.1007/978-3-031-78495-8_1

a large amount of labeled data [39]. In this sense, it is crucial to understand how difficult and time-consuming the manual labeling process is regarding different data domains [7]. Regarding 3D data such as point clouds and meshes, manual labeling is often more laborious and complex [10]. It occurs due to the sparsity and difficulty of visualizing and defining masks or bounding boxes using only multiple 2D views of the complete three-dimensional dataset [11].

Another aspect that can increase the complexity of the labeling process is the type of ground truth to be annotated. When considering the need for geometric ground truth, including the parametrization of geometric primitive patches, surface normal vectors, local curvature, and similar attributes [29], the labeling process is much more challenging for human data annotators, especially when seeking high precision [22]. In this context, manually designed 3D CAD models can be a solution to generate synthetic data with highly detailed geometric annotation, as presented by the Koch *et al.* [22] in small-scale environments, improving the generalization capacity of supervised deep learning methods. In the industrial context, the data acquisition process is also challenging due to problems related to manufacturing productivity, security during data collection, or even confidentiality agreement violations [8,18].

This work introduces LS3DS, the first large-scale industrial synthetic dataset with CAD models, meshes, point clouds, and geometric ground truth for the geometric deep learning process. Large-scale can mean that the models of the dataset are large or that the dataset has a lot of models. The present paper considers the first definition, which means that the CAD models, point clouds, and meshes are large, representing huge industrial sites with many internal structures of different shapes and sizes. Although datasets such as ShapeNet [5] and ABC [22] have a lot of models, it is also considered small-scale since the models are single parts or individual objects and do not represent large scenes with a bunch of other structures inside. Instead, LS3DS is in a similar scale of datasets applied for autonomous driving tasks, such as KITTI [9] and nuScenes [4], as well as datasets for indoor scene understanding such as SceneNN [18] and S3DIS [2].

This work contains the following main contributions:

1. **Large-Scale Industrial 3D Dataset:** we propose the LS3DS, the first large-scale synthetic dataset of 3D CAD models, meshes, and point clouds of industrial scenes, richly annotated with ground truth normals and geometric primitive surfaces and curves.
2. **Open-Source CAD Processing Pipeline:** We provide the LS3DS generation tool, a pipeline for creating point clouds and meshes with ground truth for geometric deep learning tasks from CAD models.
3. **Benchmark in Point Cloud Geometric Primitive Fitting:** We present a benchmark using the LS3DS dataset composed of state-of-the-art methods as a quantitative metric to evaluate the performance of different algorithms and the effectiveness of the proposed dataset.

2 Related Works

In 3D geometry processing, datasets are crucial assets for algorithmic development, training, and validation. Table 1 shows the main datasets with 3D data available in the literature, presenting some important characteristics that differentiate them from the proposed dataset.

Context. We proposed a large-scale dataset, meaning that the models making part of it are large and from industrial environments. In the literature, many large-scale datasets are composed of urban scenes; some focused on autonomous driving [4,9] and some on urban scene understanding for augmented reality, robotics navigation and urban planning [6,15]. Some datasets of indoor scene understanding are also available, focused on household environments [8,18] and large offices with multiple rooms and the connections between them [2].

Despite the existence of these datasets, they are not focused on industrial structures, which have different requirements and constructive aspects. In this context, the availability of a dataset plays a crucial role in the industry's digital transformation. Although the ABC dataset [22] comprises industrial context models, they present a small scale, where each model is an object of a single context and not a large scene with multiple parts.

Data Representations. Table 1 specifies three data representations (CAD models, meshes, and point clouds) that are considered important as 3D digital depiction of real-world structures. Among them, the most unusual representation of datasets is the CAD models, in which the structures are represented concerning their proposed design. Regarding meshes, in indoor environments, the RGBD reconstruction methods that were used to generate SceneNN [18] and S3DIS [2] can generate a surface representation of the observed structures as triangular meshes. Point clouds are generated directly from LiDAR sensors without a reconstruction method in urban environments. Thus, datasets of this context are not normally composed of meshes [4,6,9,15]. In Table 1, datasets such as ABC [22] and ShapeNet [5] are synthetic small-scale collections in which the available meshes are 3D designs and not data representing real worlds scenarios.

Regarding point clouds, datasets containing meshes can generate point clouds using mesh vertices or sampling strategies. However, these methods often produce point clouds that deviate from real-world data acquisition. The LS3DS dataset generates point clouds by simulating real-world drone LiDAR acquisition, accurately capturing effects like occlusion and local sparsity.

Ground Truth. Commonly, datasets of large-scale scenes are in urban context whether indoor or outdoor environments [2,4,9,15,18]. They provide semantic ground truth since the geometric ground truth is not the main important information. That is feasible since the relation between object shapes and geometric primitives is not straightforward, mainly considering scenarios in which the objects are not industrially manufactured or not even human-made. Therefore, the structures in the urban context data are hard to represent as a collection of primitive patches of parametric types, e.g., planes, spheres, cylinders, and cones. However, human-made environments such as industrial sites are mostly composed of previously CAD-designed objects and structures with a more parametrically geometric shape. On these scenarios, another fact that brings closer the relation between the geometric primitives and object shape is that the industrial components are also most often manufactured by machines that guarantee greater precision and, as a consequence, structures more similar to what was designed. Despite the higher constructive quality of urban buildings, the industry still requires the As-Built process, also known as CAD reconstruction. In this context, geometric primitive detection and shape fitting are important auxiliary tasks to dealing with this problem [33]. More recently, in the small-scale industrial context, Romanengo *et al.* [35] proposed a benchmark in point cloud geometric primitive fitting representing CAD models. Seeking to contribute to the automatic As-Built problem, datasets such as the LS3DS with geometric ground truth in large-scale context are essential.

Table 1. Overview of public datasets detailing real and synthetic ones with corresponding context, 3D data representations, and type of ground truth that each one has. Indust refers to industrial context, Geom is Geometric, and Sem is Semantic ground truths.

	Name	Context		Data Representations			Ground Truth	
	Dataset	Large-Scale	Indust.	CAD Models	Point Clouds	Meshes	Geom.	Sem.
Real	Semantic3D [15]	✓	×	×	✓	×	×	✓
	S3DIS [2]	✓	×	×	✓	✓	×	✓
	KITTI [9]	✓	×	×	✓	×	×	✓
	nuScenes [4]	✓	×	×	✓	×	×	✓
	SceneNN [18]	✓	×	×	✓	✓	×	✓
Synthetic	ABC [22]	×	✓	✓	×	✓	✓	×
	ShapeNet [5]	×	×	×	×	✓	×	✓
	STPLS3D [6]	✓	×	×	✓	×	×	✓
	Front3D [8]	✓	×	×	✓	✓	×	✓
	LS3DS	✓	✓	✓	✓	✓	✓	×

3 3D Data Generation from CAD Models

Most point cloud datasets used for deep learning are composed of real data captured in specific scenarios and manually annotated using point cloud annotation software. Although manually annotated data is common in segmentation, detection, and classification tasks, this annotation approach cannot be easily employed for geometric problems [22], such as geometric primitive fitting. In addition, especially in industrial scenarios, most companies are not interested in making 3D scans of their industrial sites openly available. Thus, it is hard to build a real-world 3D industrial dataset with sufficient models to produce generalization while training deep learning methods. To address the mentioned limitation, we present the proposed 3D data synthetic generation process.

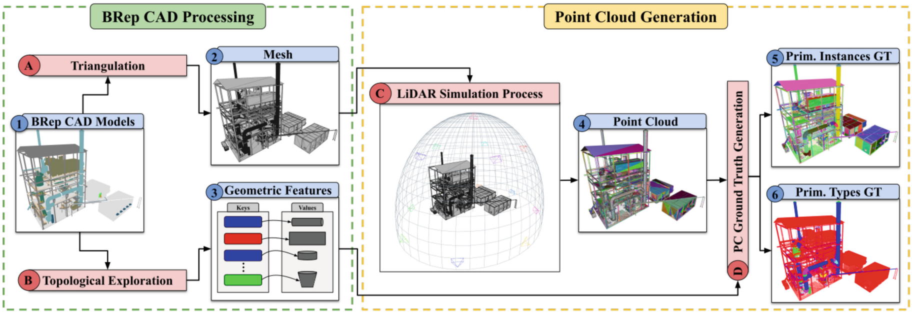


Fig. 1. 3D data generation from CAD models pipeline. First, in green, the **BRep CAD Processing** module is responsible for reading the CAD file (**1**) in STEP format, generating a mesh (**2**) and generating a features file (**3**), with the geometric primitive curve and surfaces from the input model. Second, in yellow, the **Point Cloud Generation** module applies a multi-view LiDAR simulation (**A**) approach to take the mesh and generate a point cloud (**4**). After that, in the same module, the point cloud (**4**) is associated with the geometric features (**3**) to produce the segmentation ground truth, showed instances (**6**) and as types (**6**). (Color figure online)

Our CAD processing pipeline is divided into two main steps: *i*) BRep CAD Processing and *ii*) Point Cloud Generation. The first uses a CAD model as input, generating three outputs: *i*) a mesh representing the structure; *ii*) a feature file containing the model's geometric primitives; and *iii*) a statistics file are also recorded. The second step is dedicated to point cloud generation, receiving the mesh as input, correlating the points to its geometry, and obtaining the geometric ground truth data. The methodology is shown in the diagram of Fig. 1.

BRep CAD Processing. We start from a 3D BRep CAD model in STEP [20] format, i.e., a tree data structure of geometrical entities detailed in [1]. First, a topological exploration is applied to collect the geometric curves, surfaces, and corresponding parameters. Each geometric primitive in the tree structure is explored and included in a hash table. The separate chaining approach is adopted in [23] to solve collision problems in hashing data structure. They use linked lists of geometric primitives in each hash table line. Thereby, the geometric primitives in the tree data structure of the BRep CAD model are annotated and saved into a features file, following similar patterns of ABC dataset [22]. We highlight that the choice of hashing data structure with separate chaining is due to its efficient performance, with the computational complexity of O(1).

The BRep CAD processing also carried out a stage to create a mesh from a 3D CAD model. For this, the incremental triangulation process is performed. The triangular meshes of each component are aggregated into a single mesh representing the entire CAD model. During this aggregation process, the vertices and edges in more than one geometric component local mesh are referenced with the same index in the result, preserving the watertightness property of the final mesh. Also, we keep track of the relation between the mesh and the geometric primitives based on the knowledge of the mesh associated with each vertex and face. In this way, a list of vertices and face indexes are added to the parameters of each geometric primitive in the features file.

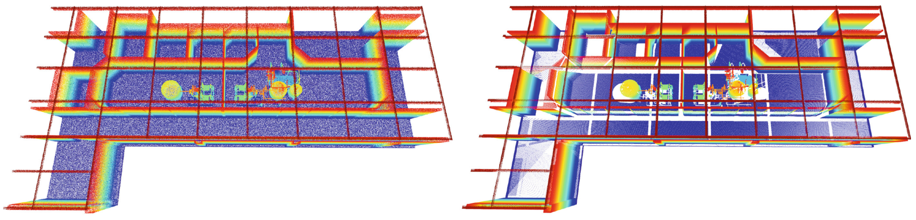
Point Cloud Generation. A widely used approach for point cloud generation involves sampling points on a mesh model. This method is particularly suitable for study cases as it accurately reconstructs all components and geometries in the 3D model. The resulting synthetic point cloud is comprehensive, capturing even hidden parts such as the interior of closed geometry structures, in which these parts should be inherently occluded. The sampling process is based on uniformly distributing a set of N points along the mesh surface. For this, each triangle with index k in the mesh receives a weight w_k according to the ratio of its area to the total mesh surface area. Thus, the number of points n_k to be sampled in each triangle is calculated from the product between the weight and the total amount of points (N) initially chosen, following the mathematical formulation: $n_k = w_k \times N$. Finally, each triangle receives n_k points following a uniform random distribution on its surface.

We propose a synthetic point cloud generation method in our pipeline by simulating LiDAR data acquisition using a multi-view scheme around the structure mesh. In this simulation, we employ a semi-ellipsoidal fitting surrounding the model with an offset distance in which the generated points. The LiDARs are strategically positioned within this ellipsoid surface to observe the structure from multiple points of view. This approach is innovative in this context and enables the generation of point clouds that reproduce some effects in a real-world LiDAR data collection scenario. In contrast to sampling approaches, the LiDAR simulation method mimics local sparsity, with fewer points in regions such as the ones far away from the simulated sensor. In addition, the most important real-world reproduced effect is the occlusion, in which regions of the structure that could not be observed from any point of view do not receive any points. With point clouds close to the real world, the methods to be trained using these synthetic data should be more easily transposed to real application contexts, reducing the sim-to-real gap.

Figure 2 presents a comparative analysis of the previously mentioned approaches. As the LiDAR simulation method has generated 2.8 million points in this structure, the same amount is used in the sampling generation. In Fig. 2a, the widely used uniform sampling method was employed to generate points in the input mesh. Meanwhile, in Fig. 2b, the point cloud was generated using the LiDAR simulation method, in which it is possible to observe regions entirely white, i.e., with nos points (occluded) and regions with a fainter color, i.e., with fewer points (sparse).

4 LS3DS Dataset

The LS3DS dataset is generated using the proposed 3D data generation pipeline, and it is composed of 77 free-to-use CAD models available online in STEP format. Although 77 models may seem small to generalize the learning process, it



(a) (b)

Fig. 2. Point clouds generated from a triangular mesh of a 3D CAD model of LS3DS dataset, both with 2.8 million points. Figure 2a corresponds to a uniform sampling-based point cloud. Figure 2b corresponds to a point cloud generated from LiDAR simulation.

is important to highlight that the models will be decomposed into thousands of parts to the learning-based methods to be trained in. Section 5 provides more details about the model decomposition. The notion of large-scale used here considers the following characteristics: wider cross-section area, total volume, and geometric primitives density, resulting in complex scenes of industrial environments. Total volume and visual analysis were used as metrics to select 3D models of industrial scenes that satisfy the geometric density characteristics and complexity of the scenes, with total volume empirically defined as 20 m^3.

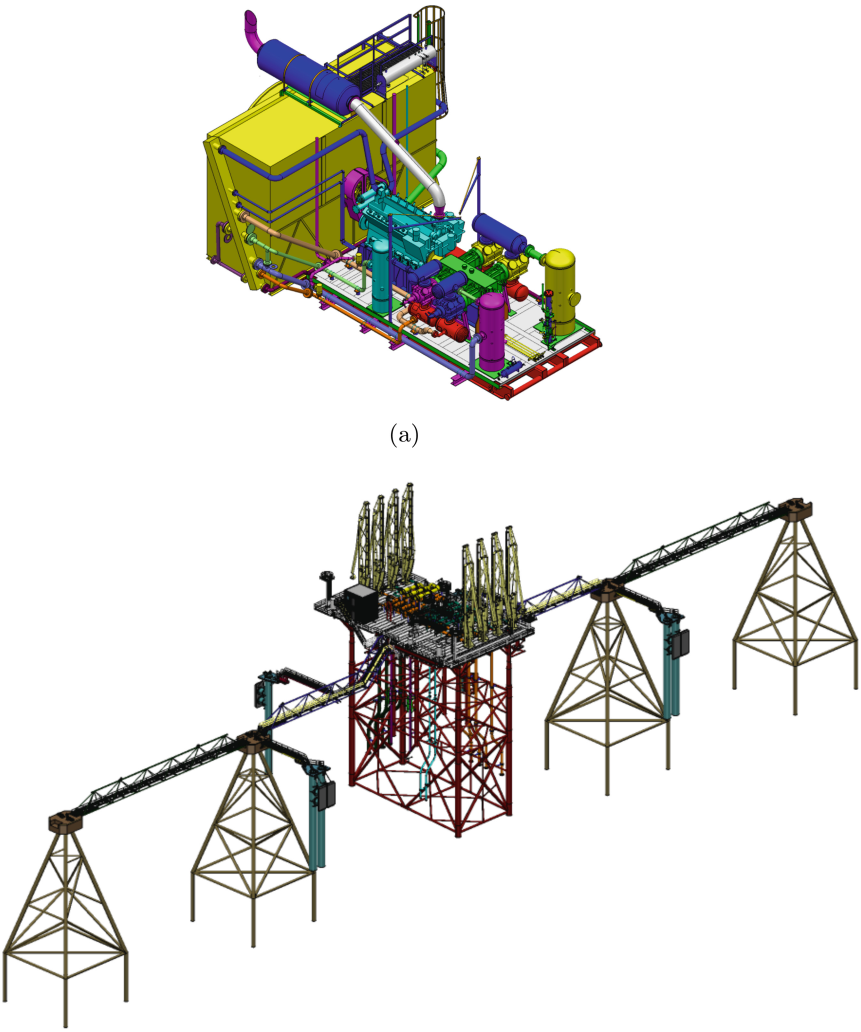
Another important feature of the dataset is that it is composed of industrial models, representing complex industrial scenes containing different objects, with many geometries of different types. Heading in the same direction of the dataset diversity, LS3DS has a wide scale range, going from 5 m^2 up to 2000 m^2 of cross-section area, as illustrated respectively in Fig. 3.

The meshes of LS3DS are used just as an intermediary representation to obtain the point clouds. Our modified Open CASCADE Technology (OCCT) library version generates the meshes. Focusing on the point clouds, the generation process uses the LiDAR simulation technique previously described. Concerning the geometries, the chart shown in Fig. 4 quantifies the average amount of each geometry type in dataset models. Figure 4b shows the planes and cylinders are the most frequent surfaces in the LS3DS Dataset, by a large margin, being 88.6%. The third most frequent type is the b-splines. Following up are the cones, torus, and spheres, summing up 7.64% of all surface entities of the dataset.

Although the charts in Fig. 4 show valuable information, the average number of instances is not the only metric to understand the distribution of LS3DS geometries. Analyzing the average surface area occupied by each type contributes to understanding what geometry types receive more points and are more important in the large-scale industrial context.

5 Benchmark in Geometric Primitive Fitting

The proposed dataset presents a challenging scenario of industrial sites with high geometry density and large-scale structures. In this context, although the

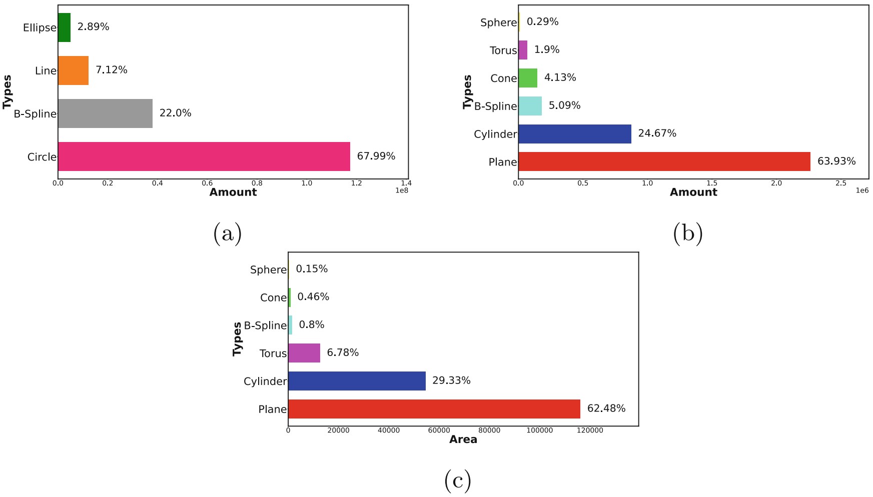


(a)

(b)

Fig. 3. Scale diversity in LS3DS Dataset. Figure 3a shows the model with the smallest cross-section area model (5 m^2). Figure 3b shows the model with the largest cross-section area model (2000 m^2).

point clouds are synthetic, their generation method intentionally produces hard-to-handle effects, such as occlusion and local sparsity, which leads to incomplete and under-sampled structures in the data. In addition to these challenges, the number of points in the clouds that represent large-scale industrial scenes is huge, as is the case with well-collected data by physical sensors. Therefore, it increases the difficulty of solving the problems established in the dataset and the



(a)

(b)

(c)

Fig. 4. Amount of geometric entities of each type in the LS3DS dataset. Figure 4a shows the distribution of the number of curves by type. Figure 4b shows the distribution of the number of surfaces by type. Figure 4c shows the distribution of the mean area of surfaces by type.

similarity with real-world scenarios even more. Because the generation pipeline already produces geometric ground truth, the LS3DS dataset can be used to train and validate methods on various 3D scene geometry understanding tasks.

In industrial scenes, which present high precision in the building process, the geometries used to design the structures are mostly well-represented in the real buildings. In addition, a proper 3D digital representation of the industrial structure as it was built is fundamental to the industry's digital transformation. As commented in [33], geometric primitive detection and fitting are key tasks to detect the building elements and generate a parametric representation, which can be used to produce As-Built models. After being exploited by deep learning methods in small-scale structures [19,24,26,28,38,41,42], the lack of a proper dataset with large-scale models is brought in PrimitiveNet [19] as an important restriction to enable the development of methods to face the problem in this context. In addition to making this data available, the present paper provides a benchmark in the problem of point cloud geometric primitive fitting applied to large-scale industrial structures.

Point Cloud Geometric Primitive Fitting Problem. Geometric primitives are a lightweight and concise representation for real-world manufactured objects or even scenes, being directly used for As-Built of large environments [30], CAD reverse engineering [14,27], point cloud simplification [13,36] or just as a regular way to represent some object parts [40]. These methods that use geometric primitives detected on point clouds for specific applications need some support approach to segment and fit those geometries that are implicitly in the points with the highest possible precision.

Point cloud geometric primitive fitting is finding a set of geometries of primitive types in the input point cloud. The mentioned problem is composed of primitive instance segmentation and parameter fitting sub-problems. This problem can be defined as a chicken-and-egg problem [25] since if the primitive parameters are known; the nearest point-to-primitive distance can be used to determine the membership between point and primitives. Additionally, robust parameter fitting methods can be used to compute the set of parameters for each primitive, considering points segmentation in geometry instances as known. This approach is adopted in the model estimation module of SPFN [26] or in RANSAC-based approaches used in Efficient RANSAC (ERANSAC) [36].

As shown by Kaiser *et al.* [21], the classical methods (i.e., non-learning-based) to solve the problem is mostly based on theoretical foundations such as RANSAC, parameter space, and primitive growing. Among them, the ERANSAC [36] is the most widely used approach. However, after SPFN [25], various learning-based methods have been proposed. SPFN used a PointNet++ [34] first to predict per-point properties and after fit geometric primitives using a differentiable module. CPFN [24] improved the SPFN results, mainly in high-resolution point clouds, using a cascade network with a global and a local SPFN. ParseNet [38], HPNet [42] and QuadricsNet [41] used backbone to build an embedding space of higher dimension and applied a mean-shift algorithm to cluster the point cloud into geometric primitive instances. PrimitiveNet [19] used a local-based approach to receive not just the points but the neighboring edges between them as input. SED-Net [28] and ComplexGen [14] have improved the quality of their results by predicting geometric surfaces and curves.

Geometric primitive fitting is a problem similar to the widely explored point cloud instance segmentation research field [31,37,43]. The problem can be defined as identifying and separating individual parts in the input point cloud [16]. However, the instance segmentation problem of geometric primitive fitting demands each instance to receive class labels of geometric primitive types, unlike the semantic classes such as road, building, and car from urban datasets [4,9] or such as wall, ceiling, and chair from indoor datasets [2,18]. Although methods like ParseNet [38] and HPNet [42] segment and fit B-Splines, the typical subset of types covered by geometric primitive fitting methods are plane, sphere, cylinder, and cone. Using LS3DS as a reference, Fig. 4c shows that just these four types cover 92.4% of the total area of the dataset structures, stating that, although not ideal, it is a suitable subset of types to represent these industrial sites.

Metrics. To evaluate the methods in the task using the ground truth dataset, the same metrics defined by SPFN [25] are used, which are:

- ***Segmentation Mean Intersection over Union (SIoU):*** mean correspondence between detected primitive instances and the same ground truth instances.

- ***Mean Primitive Type Accuracy (TAcc):*** mean accuracy of primitive instances type attribution.
- ***Residual Error (Res):*** mean point-to-primitive (P2P) error, in which the distance of each point that belongs to a primitive on the ground truth instance mask is measured to the predicted primitive.
- ***P Coverage (PCov)***[1]: mean percentage of points from predicted instance mask that is closer than a pre-defined ε threshold to its primitives.

Problem of Learning in Large-scale Environments. Although the geometric primitive fitting problem is faced by deep learning techniques in small-scale contexts, the problem's difficulty increases as the scale of the input structures increases. In large-scale contexts, mainly in industrial sites, the construction aspects differ greatly from small-scale individual objects, where state-of-the-art deep learning-based methods are normally trained and validated. This aspect can be explained by understanding the differences in the interconnection of internal structures in small-scale versus large-scale models. In small-scale, such as ABC Dataset [22] models, the internal parts of the objects are geared more towards the complete connectivity, where the geometries are recursively related to each other in a fully connected graph topology. On the other hand, scenes are more complex in large-scale models, considering that many individual objects inside them are unrelated to all the others. In this way, it is possible to understand that while small-scale structures are normally single-context objects, large-scale ones are multi-context models with many small parts that are not necessarily related to each other. In addition to the diversity of contexts, the scale of each part can be a lot different since at the same time, a large-scale structure can have large floor or wall planes and small valves or motors, making learning in large-scale environments even more difficult.

Beyond the fact that solving problems in large-scale environments is harder than in small-scale environments because of many structural composition factors, The point clouds that represent a large model in a good resolution have a greater amount of points. In this way, considering the higher dimension of the raw data, the learning process becomes even harder since the neural network architectures must compute features for more input points, heading in the direction of the curse of dimensionality, explicated in the field of geometric deep learning [?]. In this sense, semantic and instance segmentation methods prioritize the memory and computational efficiency of the methods for them to be used in large-scale contexts. RandLA-Net [17] is an efficient example of a semantic segmentation problem in which the lightweight network architecture enables the process of 1 million points in a single pass. Other approaches use voxelization and sparse convolutions [12] to process the point cloud as a voxel grid 3D to do semantic segmentation efficiently. Although methods for large-scale contexts must be computationally efficient, some datasets are huger, and their scenes are too large to process each point cloud in a single pass. Thus, the split and merge approach is used to process then [3,37]. Inspired by this, this benchmark will adapt the deep learning-based methods using a grid-based split and merge

[1] For this benchmark, we consider $\varepsilon = 0.05$.

approach, making each large point cloud a bunch of small ones to be processed separately to generate the results to be merged afterward.

The split step uses a cuboid-based approach to split each dataset model into regions of size 2 × 2 × 2 m. Thereby, the number of models for the learning process increases with thousands of models in the training and validation stages, improving the generalization, as seen in Table 2. Additionally, it is worth mentioning that the split and merge approach can produce a loss of context, due to dividing the model into parts. However, when dividing geometric primitives into two or more parts, the parts remain surfaces described by the type of original geometric primitive and the respective equation. In this sense, the geometric primitive fitting task is less impacted than other tasks, such as semantic and instance segmentation.

Table 2. Number of models in the dataset without splitting and splitting it. The size used in the paper benchmark for the cuboid split was 2 × 2 × 2.

Dataset	Mean Vol. (m^3)	Train Size	Val. Size
Complete Data	≈ 47376	52	25
Split w/ 8 × 8 × 8	512	1124	755
Split w/ 4 × 4 × 4	64	8731	2753
Split w/ 2 × 2 × 2	**8**	**69626**	**6908**

Baseline for Comparisons. ERANSAC [36] is the most adopted method to detect and fit parameters of multiple types of geometric primitives to point clouds since it is used as baseline by the small-scale methods [14, 19, 26, 28, 38, 42]. Although Kaiser *et al.* [21] classify ERANSAC as a method for individual objects context, this approach has no explicit points limitation, and it is evaluated with point clouds of about 2 million points in [36]. In this way, the present benchmark considered ERANSAC as the baseline for comparing deep learning-based strategies. To be fair and study the impact of the split and merge procedure in the results, a modified version of ERANSAC was also evaluated.

Geometric Deep Learning Methods. Relating to the deep learning-based methods, the split and merge adaptation is applied to make them able to process large-scale point clouds. Even with this approach, some methods are not suitable for the benchmark . SPFN [25] was not used because of the inflation of the memory consumption when increasing the hyper-parameter K, and because of some other constraints that hindered the large-scale adaptation of the official code[2]. CPFN [24] uses two SPFN networks inside its architecture, holding the

[2] The SPFN [25] official code assumes that all the geometric primitive instances in a model must have the same number of points.

same challenges in the K hyper-parameter setting and memory consumption. PrimitiveNet [19] demands a mesh directly transformed to a point cloud as input, which is not possible given the fact that the LS3DS point clouds were generated using a LiDAR simulation approach[3]. ComplexGen [14] was not used because the main subject of the method is to directly reconstruct the CAD model using a B-Rep Chain Complex Generation and not just fit geometric primitives to point clouds, requiring additional ground truth. Despite that, the ComplexGen is already heavy[4] to be trained in small-scale and it predicts the wireframe of the objects, which may have conflicts split-and-merge approach adopted in this benchmark. SED-Net [28] did not a pubnovailable code at the time of this benchmark's development. Thus, the deep learning-based approaches compared with the classical baseline are ParseNet [38] and HPNet [42].

Despite the split and merge adaptation, the original methods are focused on small-scale structures, which means that the methods do not aim to process many points in a single pass. The subset of ABC dataset [22] adopted by ParseNet [38] has $10K$ points for each model, and that is the amount that the method can process. Even though HPNet [42] uses the same subset, an additional random sample filter is used to reduce this amount to $7K$ points. Thus, aiming at a fair comparison, every method was trained and validated in a dataset where each set has $7K$ points, which is the least amount that every method of this benchmark can directly take in a single pass.

Benchmark Discussion. Table 3 shows the quantitative results of the proposed benchmark. The methods with * are the versions adapted to large-scale using the split and merge methodology and some other specific required modifications for each method. After the split, the generated dataset is merged again to produce a version to be processed by the non-modified version of ERANSAC [36], making the baseline run in the same points with the same primitives on the ground truth as the divided version. All deep learning-based methods on this benchmark were used with the default parameters of their official implementations. For the baseline, the implementation of ERANSAC on CGAL [32] library was used, using the default parameters as well.

Comparing ParseNet* [38] and HPNet* [42] with ERANSAC and ERANSAC*, it is possible to note that the deep learning-based methods have better results in almost every metric. Concerning the SIoU, which evaluates the quality of instance segmentation, the methods modified by the split and merge approach have taken advantage. Looking at the qualitative results in Fig. 5, it is possible to understand that processing small parts separately improves finding small details that divide similar primitives. This fact can be observed by com-

[3] The PrimitiveNet [19] authors suggestion of using a KNN to generate the neighbor graph between point did no produce feasible results.

[4] The ComplexGen [14] method takes three days o 8 NVIDIA V100 GPUs to be trained for small-scale context.

Table 3. Quantitative results of the methods ERANSAC, ERANSAC*, ParseNet* and HPNet* in point cloud geometric primitive fitting on LS3DS dataset. We used the metrics SIoU, TAcc, Res, and PCov for this.

Method	↑ SIoU	↑ TAcc	↓ Res	↑ PCov
ERANSAC [36]	0.2150	0.3627	0.0424	0.5340
ERANSAC* [36]	0.5528	0.4625	0.0111	**0.9917**
ParseNet* [38]	0.5633	0.9229	0.0127	0.9461
HPNet* [42]	**0.5774**	**0.9233**	**0.0070**	0.9886

paring ERANSAC and ERANSAC* results. Even though a significant part of the improvements in SIoU are from the split approach, the deep learning-based methods can produce better results with relation to the quality of instance segmentation, with emphasis on the HPNet* [42], which achieved the best quantitative and qualitative results.

Regarding accuracy in type attribution for each instance, the metric TAcc presents the most different results between the baseline and the deep learning-based methods. On this metric, which evaluates the percentage of instances with the right attributed type, the learning-based methods achieved two times better results than the classical approaches. This can be explained by the fact that the only heuristic that RANSAC-based approaches use to evaluate if a primitive is suitable for a set of points is by looking at the percentage of them covered by that primitive. On the other hand, learning-based methods extract information from the training set of dataset LS3DS. Thus, as there is no restriction, extremely large cones and cylinders are constantly fitted to represent plane surfaces.

Through the proposed benchmark, mainly because of better results of deep learning-based methods, it is possible to state that the LS3DS dataset is useful and important for the large-scale geometric deep learning research. In addition, some limitations of the state-of-the-art techniques can be disclosed since its network architectures' computational and memory non-efficiency lead to a stronger use of split and merge approaches. This characteristic leads to the need to use fewer points in each divided part and, as a consequence, the split of the input cloud in more parts to keep a good resolution in the points. Although the results are promising, it is expected that, through the use of the LS3DS dataset, new techniques for geometric deep learning on large-scale 3D data will be developed using the knowledge of other research fields, such as 3D semantic segmentation and 3D instance segmentation.

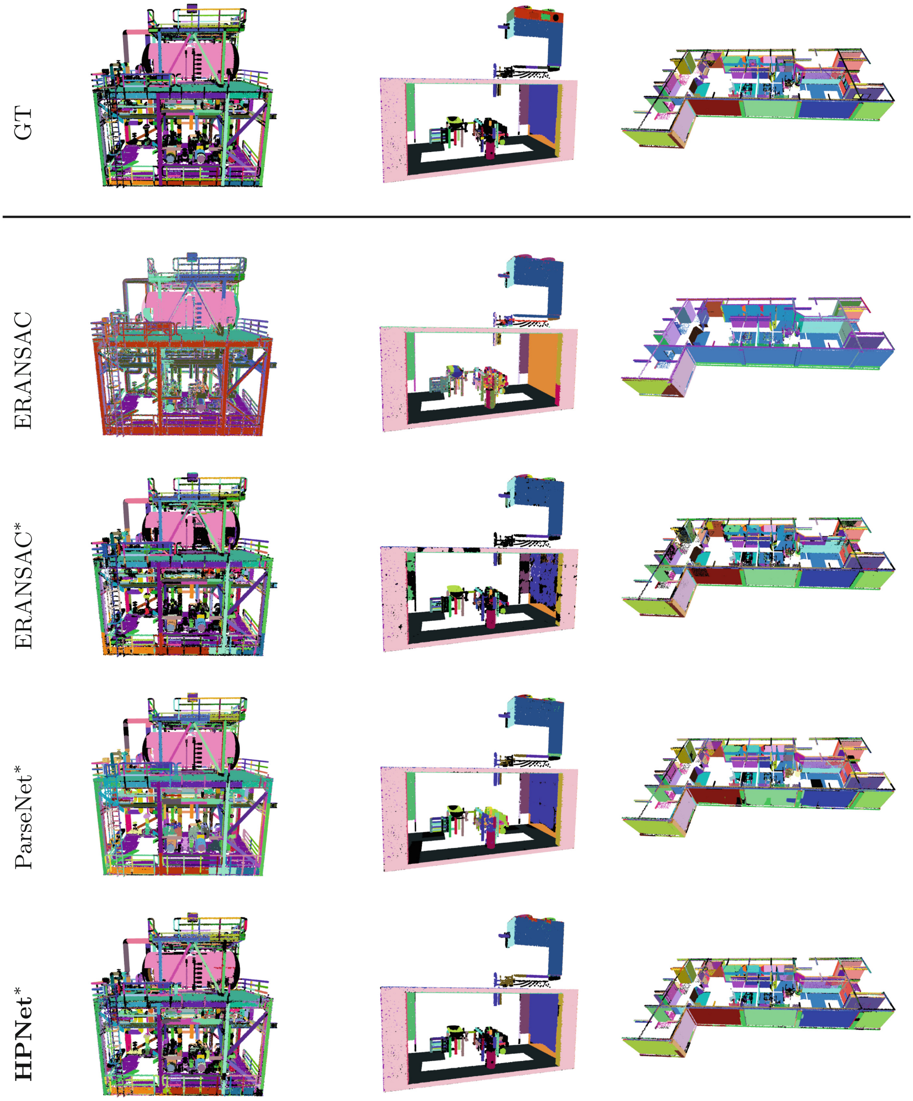


Fig. 5. Qualitative primitive instance segmentation results of the methods on three models of LS3DS dataset. In the qualitative analysis were used ERANSAC, ERANSAC*, ParseNet* and HPNet* methods. In the results, each color represent a different predicted primitive patch of the types plane, cylinder, sphere and cone. The first row shows the LS3DS ground truth and the following rows show the results of each evaluated method.

6 Conclusion

We present the LS3DS, a dataset of large-scale industrial scenes with 3D data as CAD models, meshes, and point clouds with ground truth that can be used in many geometric deep-learning tasks. The LS3DS generation tool is provided with the dataset, enabling the dataset expansion or switching of the context of the input CAD models to generate other datasets. It is important to highlight that the point clouds are generated through a multi-view LiDAR simulation method, which reproduces effects such as occlusion and local sparsity in the produced data, closing the gap between the generated synthetic data and the real-world ones.

To validate the dataset's importance, a benchmark in large-scale point cloud geometric primitive fitting is provided. In this context, the deep learning-based methods have reached the best results, emphasizing the HPNet [42], the best method of this benchmark. Although the results are promising, the adaptations of the deep learning-based methods to make them able to run on a large scale lead to a high information loss, which keeps the problem open for proposing new and more scalable methods. Beyond the benchmark, LS3DS can be used in other 3D scene understanding problems in which its robustness, quality, and efficiency are required. Finally, the LS3DS can be easily expanded to another context or fitted to a more specific one, where not only will the dataset be made openly available, but the code of the processing pipeline is available. Thereby, anyone can generate an expanded version of LS3DS or a new version in another context just by changing the CAD models in the input of the data generation pipeline. Future works will focus on working with industrial partners to make available real-world scans and their associated 3D CAD.

Acknowledgements. This study was funded by the Human Resource Program of The Brazilian National Agency for Petroleum, Natural Gas, and Biofuels - PRH-ANP, supported by resources from oil companies considering the contract clause no 50/2015 of R, D&I of the ANP. The authors also thank the Aeronautics Institute of Technology (ITA), the National Council for Scientific and Technological Development (CNPq), and the Study and Project Funding Agency (Finep).

References

1. Al-wswasi, M., Ivanov, A.: A novel and smart interactive feature recognition system for rotational parts using a step file. Int. J. Adv. Manuf. Technol. **104**(1), 261–284 (2019)
2. Armeni, I., Sax, S., Zamir, A.R., Savarese, S.: Joint 2d-3d-semantic data for indoor scene understanding. arXiv preprint arXiv:1702.01105 (2017)

3. Boulch, A.: Convpoint: continuous convolutions for point cloud processing. Comput. Graph. **88**, 24–34 (2020)
4. Caesar, H., et al.: Nuscenes: a multimodal dataset for autonomous driving. In: IEEE/CVF CVPR, pp. 11621–11631 (2020)
5. Chang, A.X., et al.: Shapenet: an information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
6. Chen, M., et al.: Stpls3d: a large-scale synthetic and real aerial photogrammetry 3d point cloud dataset. In: The British Machine Vision Conference (BMVC), BMVA Press (2022)
7. Dos Santos, M.M., De Giacomo, G.G., Drews-Jr, P.L., Botelho, S.S.: Matching color aerial images and underwater sonar images using deep learning for underwater localization. IEEE RA-L **5**(4), 6365–6370 (2020)
8. Fu, H., et al.: 3d-future: 3d furniture shape with texture. Int. J. Comp. Vis. **129**, 3313–3337 (2021)
9. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: the kitti dataset. Int. J. Robot. Res. (IJRR) (2013)
10. Gosala, N., Petek, K., Drews-Jr, P.L., Burgard, W., Valada, A.: Skyeye: self-supervised bird's-eye-view semantic mapping using monocular frontal view images. In: IEEE/CVF CVPR, pp. 14901–14910 (2023)
11. Gosala, N., et al.: Letsmap: unsupervised representation learning for semantic bev mapping. arXiv preprint arXiv:2405.18852 (2024)
12. Graham, B., van der Maaten, L.: Submanifold sparse convolutional networks. arXiv preprint arXiv:1706.01307 (2017)
13. Guinard, S.A., Daniel, S., Badard, T.: 3d point clouds simplification based on geometric primitives and graph-structured optimization. In: 2022 26th International Conference on Pattern Recognition (ICPR), pp. 3837–3844. IEEE (2022)
14. Guo, H., Liu, S., Pan, H., Liu, Y., Tong, X., Guo, B.: ComplexGen: CAD reconstruction by B-rep chain complex generation. ACM SIGGRAPH **41**(4) (2022)
15. Hackel, T., Savinov, N., Ladicky, L., Wegner, J.D., Schindler, K., Pollefeys, M.: SEMANTIC3D.NET: a new large-scale point cloud classification benchmark. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. IV-1-W1, pp. 91–98 (2017)
16. Hafiz, A.M., Bhat, G.M.: A survey on instance segmentation: state of the art. Int. J. Multimedia Inf. Retrieval **9**, 171–189 (2020). https://doi.org/10.1007/s13735-020-00195-x
17. Hu, Q., et al.: Randla-net: Efficient semantic segmentation of large-scale point clouds. In: IEEE/CVF CVPR, pp. 11108–11117 (2020)
18. Hua, B.S., Pham, Q.H., Nguyen, D.T., Tran, M.K., Yu, L.F., Yeung, S.K.: Scenenn: a scene meshes dataset with annotations. In: International Conference on 3D Vision (3DV), pp. 92–101. IEEE (2016)
19. Huang, J., Zhang, Y., Sun, M.: Primitivenet: primitive instance segmentation with local primitive embedding under adversarial metric. In: IEEE/CVF ICCV, pp. 15323–15333 (2021). https://doi.org/10.1109/ICCV48922.2021.01506
20. International Organization for Standardization (ISO): Industrial automation systems and integration - Product data representation and exchange -Part 203: Application protocol: Configuration controlled 3D designs of mechanical parts and assemblies. Technical Report, International Organization for Standardization, December 1994
21. Kaiser, A., Ybanez Zepeda, J.A., Boubekeur, T.: A survey of simple geometric primitives detection methods for captured 3d data. In: Computer Graphics Forum, vol. 38, pp. 167–196. Wiley Online Library (2019)

22. Koch, S., et al.: Abc: a big cad model dataset for geometric deep learning. In: IEEE/CVF CVPR, June 2019
23. Köppl, D.: Separate chaining meets compact hashing. arXiv preprint arXiv:1905.00163 (2019)
24. Lê, E.T., Sung, M., Ceylan, D., Mech, R., Boubekeur, T., Mitra, N.J.: Cpfn: cascaded primitive fitting networks for high-resolution point clouds. In: IEEE/CVF ICCV, pp. 7457–7466, October 2021
25. Li, D., Feng, C.: Primitive fitting using deep geometric segmentation. In: ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction, vol. 36, pp. 780–787. IAARC Publications (2019)
26. Li, L., Sung, M., Dubrovina, A., Yi, L., Guibas, L.J.: Supervised fitting of geometric primitives to 3d point clouds. In: IEEE/CVF CVPR, pp. 2652–2660 (2019)
27. Li, Y., Wu, X., Chrysathou, Y., Sharf, A., Cohen-Or, D., Mitra, N.J.: Globfit: consistently fitting primitives by discovering global relations. ACM Trans. Graph. **30**(4) (2011). https://doi.org/10.1145/2010324.1964947
28. Li, Y., Liu, S., Yang, X., Guo, J., Guo, J., Guo, Y.: Surface and edge detection for primitive fitting of point clouds. In: ACM SIGGRAPH 2023 Conference Proceedings, pp. 1–10 (2023)
29. Maurell, I.P., et al.: Volume change estimation of underwater structures using 2-d sonar data. IEEE Sens. J. **22**(23), 23380–23392 (2022)
30. Murali, S., Speciale, P., Oswald, M.R., Pollefeys, M.: Indoor scan2bim: building information models of house interiors. In: IEEE/RSJ IROS, pp. 6126–6133. IEEE (2017)
31. Ngo, T.D., Hua, B.S., Nguyen, K.: Isbnet: a 3d point cloud instance segmentation network with instance-aware sampling and box-aware dynamic convolution. In: IEEE/CVF CVPR, pp. 13550–13559 (2023)
32. Oesau, S., et al.: Shape detection. In: CGAL User and Reference Manual. CGAL Editorial Board, 5.6 edn. (2023)
33. Pătrăucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., Haas, C.: State of research in automatic as-built modelling. Adv. Eng. Info. **29**(2), 162–171 (2015)
34. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: deep hierarchical feature learning on point sets in a metric space. Adv. in neural Info. Process. Syst. **30** (2017)
35. Romanengo, C., Raffo, A., Qie, Y., Anwer, N., Falcidieno, B.: Fit4CAD: a point cloud benchmark for fitting simple geometric primitives in cad objects. Comput. Graph. **102**, 133–143 (2022)
36. Schnabel, R., Wahl, R., Klein, R.: Efficient ransac for point-cloud shape detection. In: Computer Graphics Forum, vol. 26-2, pp. 214–226. Wiley Online Library (2007)
37. Schult, J., Engelmann, F., Hermans, A., Litany, O., Tang, S., Leibe, B.: Mask3d: mask transformer for 3d semantic instance segmentation. In: 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 8216–8223. IEEE (2023)
38. Sharma, G., Liu, D., Maji, S., Kalogerakis, E., Chaudhuri, S., Měch, R.: Parsenet: a parametric surface fitting network for 3d point clouds. In: European Conference on Computer Vision (ECCV), pp. 261–276. Springer (2020)
39. Steffens, C.R., Messias, L.R., Drews-Jr, P.L.J., Botelho, S.S.C.: CNN based image restoration: Adjusting ill-exposed srgb images in post-processing. J. Intell. Robot. Syst. **99**(3), 609–627 (2020)
40. Vasu, S., Talabot, N., Lukoianov, A., Baqué, P., Donier, J., Fua, P.: Hybridsdf: combining deep implicit shapes and geometric primitives for 3d shape representation and manipulation. In: International Conference on 3D Vision (2022)

41. Wu, J., Yu, H., Yang, W., Xia, G.S.: Quadricsnet: learning concise representation for geometric primitives in point clouds. arXiv preprint arXiv:2309.14211 (2023)
42. Yan, S., Yang, Z., Ma, C., Huang, H., Vouga, E., Huang, Q.: Hpnet: deep primitive segmentation using hybrid representations. In: IEEE/CVF CVPR, pp. 2753–2762 (2021)
43. Zhao, W., Yan, Y., Yang, C., Ye, J., Yang, X., Huang, K.: Divide and conquer: 3d point cloud instance segmentation with point-wise binarization. In: IEEE/CVF ICCV, pp. 562–571 (2023)

Visual Question Answering with Cascade of Self- and Co-Attention Blocks

Aakansha Mishra[1(✉)], Ashish Anand[1], and Prithwijit Guha[2]

[1] Department of Computer Science and Engineering, IIT Guwahati, Guwahati, India
{ak.kkb,anand.ashish}@iitg.ac.in
[2] Department of Electrical Engineering, IIT Guwahati, Guwahati, India
pguha@iitg.ac.in

Abstract. Recent advancements in Visual Question Answering (VQA) have been driven by the integration of complex attention mechanisms. This work introduces a novel approach aimed at enhancing multi-modal representations through dense interactions between visual and textual inputs in an alternating fashion. The proposed model features an attention framework that incorporates both self-attention and co-attention mechanisms, strategically applied to image and text modalities. Self-attention modules capture contextual dependencies among objects in images and words in questions, crucial for accurate inference of answers. Meanwhile, co-attention mechanisms facilitate effective cross-modal interactions between images and text. To extract fine-grained information from both modalities, we introduce a Cascade of Self- and Co-Attention blocks (CSCA). This architecture is evaluated extensively on prominent benchmarks including VQA2.0, TDIUC, and GQA datasets. Experimental results, including comprehensive ablation studies, highlight the effectiveness of the model's key components and the cascading nature of attention mechanisms in enhancing performance across diverse VQA tasks.

Keywords: VQA · Attention · Self-Attention · Co-attention · Multi-modal Fusion · Classification Networks

1 Introduction

Visual Question Answering [1,2,10] is a challenging multimodal AI task that aims to answer a natural language question about an image context. VQA task has captured considerable interest within the vision and language research community due to its extensive practical applications, including assisting the visually impaired, advancing autonomous vehicles, refining visual chatbots etc. Simultaneously, VQA is a challenging multimodal task that requires a deep semantic understanding of images to predict answers accurately. Achieving this requires seamlessly integrating visual and linguistic information, leveraging intricate reasoning capabilities, and deploying advanced attention mechanisms for effective question-answering.

Since seminal works such as [2], significant efforts have been directed towards enriching the representation of text and image modalities. These efforts have primarily focused on advancing beyond simple fusion-based features to more sophisticated attention-based approaches [1,7,8,30,31,38,39]. Early attention-based

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 20–36, 2025.
https://doi.org/10.1007/978-3-031-78495-8_2

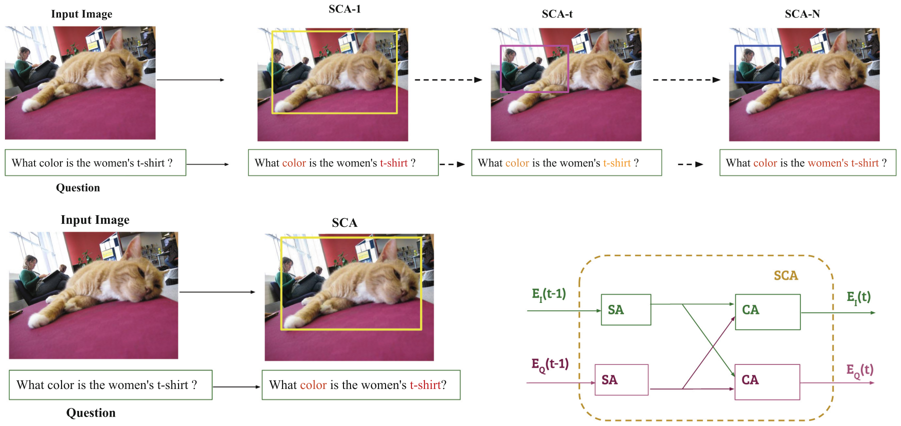

Input Image
SCA-1
SCA-t
SCA-N
What color is the women's t-shirt ?
Question
What color is the women's t-shirt ?
What color is the women's t-shirt ?
What color is the women's t-shirt ?
Input Image
SCA
What color is the women's t-shirt ?
Question
What color is the women's t-shirt?
SCA
E_I(t-1)
SA
CA
E_I(t)
E_Q(t-1)
SA
CA
E_Q(t)

Fig. 1. An example to illustrate the relevance of proposed module.[Top] An example to illustrate the relevance of attention to dual modality through the cascaded SCA module. [Bottom Left] Without cascaded attention, it is not getting refined and hence is unable to give better attention. [Bottom Right] Overview of the proposed model. An attention block, referred to as SCA, comprises of *self-attention (SA)* and *co-attention (CA)* modules. Multiple such attention blocks are cascaded, where the outputs of $(t-1)^{th}$ block ($E_I(t-1)$ and $E_Q(t-1)$) are presented as input to the $(t)^{th}$ block.

methods [1,38,39] initially concentrated on identifying salient regions within images based on the textual content of questions. This approach, termed visual attention, aimed to highlight relevant image regions. Subsequent advancements introduced co-attention methods [3], which extended this concept by integrating textual attention with visual attention. Co-attention methods proved effective in enhancing the performance of VQA systems by jointly attending to both the textual and visual modalities, focusing on relevant words within the context of the image.

A single site on image and question is insufficient to grasp the image's intricacies in the context of the question. It requires a consecutive focus on the image and multiple reads of questions to answer. Along with these requirements, if within modality, information is encoded and refined before the interaction, it will give an improved representation of context and cues. To accomplish it, this work proposes an end-to-end network with cascaded self- and co-attention blocks. This helps obtain enriched representations, leading to improved performance. Based on the advantages of each of the following modules: self-attention (SA), co-attention (CA) and a cascade of attention mechanisms, this work proposes combining them systematically. Towards this objective, the proposed model builds one self- and co-attention-based attention block (SCA) that combines both SA and CA in a specific way. For both text and image modalities, a specific SA module obtains a feature representation for the respective modality. Then, the co-attention module uses a self-attended representation of one modality and attends (takes attention) to the self-attended representation of the other modality to obtain a cross-modality contextual representation for the second modality. Thus, there are two SA modules (one for each text and image modalities) and two co-attention modules within a single SCA block (Figure 1, bottom-right). In one complex attention block of SCA, both modalities guide themselves to capture internal correlation and each other to learn the robust representation of each of the visual and textual domains.

Figure 1 (top) serves as an illustrative example to highlight the impact of the cascaded Self and Cross-Attention (SCA) module within the model to obtain an enriched representation. In the initial SCA block, the model's attention is directed towards a broad range of image regions, encompassing various objects such as 'cat,' 'women,' 'window,' and others. Additionally, it focuses on specific words like 'color' and 't-shirt,' as indicated by their attention scores. With the inclusion of multiple SCA blocks, notably after the t^{th} block, the model's attention gradually refines, shifting towards more concentrated image regions. This transition is accompanied by changes in word attention scores. Ultimately, in the final SCA block, the model's attention is concentrated on the most salient image region within the context of the given question. Simultaneously, the attention mechanism for the question becomes finely tuned to the most pertinent words that enable accurate responses.

When contrasting the results with and without the SCA module in Fig. 1 (top), it becomes evident that more than a single round of attention may be needed to capture all the relevant image regions and question words effectively. The cascaded SCA module contributes to a progressive and refined attention process, enhancing the model's ability to grasp contextual information and produce more accurate answers. The contribution of proposed framework are summarized:

- Proposed a VQA framework that employs a dense alternate attention mechanism. This framework comprises cascaded attention blocks strategically designed to refine the features extracted from visual and textual inputs iteratively.
- The iterative refinement process through a cascade of attention blocks enhances the model's ability to capture intricate details and relationships between the visual and textual data, ultimately improving its performance in answering complex questions about images.
- The core of each attention block consists of self-attention and co-attention so that the two modalities guide each other to obtain an enriched representation.
- Extensive performance evaluation along with ablation analysis of the proposed model on the three benchmark datasets – *VQA2.0* [10], *TDIUC* [15] and *GQA* [13].

2 Related Work

VQA, being a multimodal task, requires an unified representation of the text and image modalities. Initial VQA models [2,10,11,32] adopted simple fusion based approaches. These models first obtained feature representations of individual modality using corresponding pre-trained networks and then combined them to obtain a joint representation using a fusion schema. Simple fusion schemes include concatenation or element-wise summation or multiplication. Fukui et al. [6] proposed bi-linear pooling to capture interaction of components of the two modalities in a better way. Seeing the advantage of the bilinear pooling based fusion methods, further variants of bilinear pooling with lesser complexity or faster convergence were proposed. MFB [42], MLB [17], MFH [43] were proposed to obtain a representation providing better interaction of the two modalities.

Introduction of attention mechanism in equipped neural models with a systematic procedure to assign relative weights of importance to sequential inputs. Shi et al. [30] have introduced image attention guided by question to focus on salient image regions relevant to the given question. This helped in obtaining improved feature representations. This led to the development of several attention based approaches for VQA [1,16,22,36,38,39]. Studies in [22,36,39] have shown that applying attention multiple times helps in obtaining enriched representation embedded with fine-grained information.

Yu et al. [42] have proposed that attention on textual features in context of visual features along with visual attention plays a key role in VQA models. Such two way attention mechanism is referred to as *dual attention* or *co-attention* or *cross-modality attention* in the literature. We have also used these terms interchangeably. Kim et al. [16] have proposed bilinear interaction based attention for dual modality.

Another class of attention mechanism [7,8,18,21,23,35] uses intra-modal attention (self-attention) along with cross-modal attention (co-attention) to learn better feature representation. Gao et.al. [7] have proposed DFAF that uses dynamic intra-modality attention flow. Dynamical flow allows for adaptive modulation of the target modality's attention and helps in obtaining improved fusion of multimodal features. Multi-modal Latent Interaction (MLIN) [8] used multi-modal reasoning through summarization, interaction, and aggregation. Yu et al. [41] have proposed an encoder-decoder based dense attention mechanism. These models are relatively dense than the previous approaches and hence, are referred to as dense attention based models. Authors in [20,33] have proposed transformer based attention models for multimodality tasks. These models are pretrained for multiple tasks on huge datasets, that could be further exploited for downstream tasks.

Graph-based approaches, such as NSM [12,45] and XNM [28] leverage the structure of the connections between visual elements along with text data to facilitate reasoning in image analysis. These methods exploit the inherent relationships to enable more effective and meaningful reasoning. XKI [44] takes a step further by incorporating external knowledge and by integrating high-order relational attention, thereby leading to improved reasoning capabilities. OCCAM [35] specifically focused on concept induction, aiming to identify and understand concepts and their hierarchical relationships in visual reasoning tasks. Recently, Zhu et al. [46] proposed a concise and efficient dual-decoder

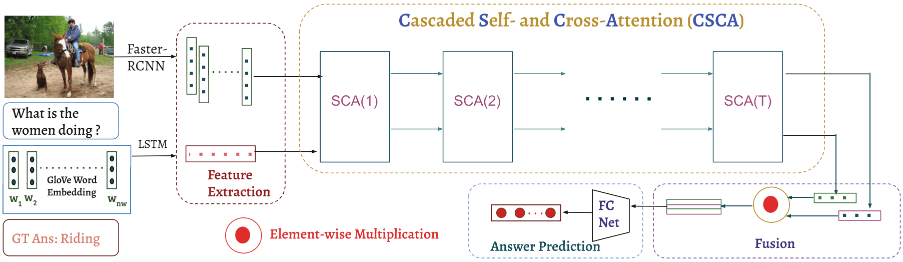


Fig. 2. Functional block diagram of the proposed approach. Initial feature extraction stage is followed by a cascade of self-attention and co-attention mechanisms. Final attended features are fused through element-wise multiplication and are fed to a fully connected network for answer classification.

Transformer network that predicts answers and provides visualized evidence, combining both linguistic and visual features.

The proposed model falls under the category of dense attention-based methods. Most inter- and intra-modal attention-based methods primarily focus on applying self-attention to the text modality only [7,8,41]. Subsequently, cross-attention is applied to the visual modality based on the self-attended text representation. However, in the proposed CSCA method, cross-modal attention is applied alternately on both modalities after the self-attention stage on dual-modality. Here, each attention block comprises intra-modality and cross-modality interactions. Unlike some existing inter- and intra-modal attention-based models, which are trained on a massive amount of data for multiple tasks, CSCA is trained from scratch and still persists in competitive performance. The proposed method is described next.

3 Proposed Method

The proposed framework treats VQA as an answer classification task following existing works like [1,2,7,8,10]. The input image I ($I \in \mathcal{I}$) and the associated natural language question q ($q \in \mathcal{Q}$) are first subjected to feature extraction (Subsect. 3.1). Pretrained deep networks [1] are used to extract features from a few salient image regions. The network embeddings are used to represent the input image. Similarly, a pretrained network is used to obtain the word embeddings of the associated input question. These word embeddings collectively represent the input question. The feature embeddings of both image and text modalities are subjected to self-attention mechanism (Subsect. 3.2) for capturing the relationships among different regions of I and words of q. The self-attended representations of these two modalities are further processed by co-attention modules (Subsect. 3.3). This single stage of **S**elf and **C**o-**A**ttention mechanism cascade forms a single SCA block (Fig. 1, bottom left). Multiple SCA blocks are cascaded to obtain further fine grained representations of both modalities.

The embeddings obtained from the final SCA block are fused (Subsect. 3.4) and fed to the answer classification network (Subsect. 3.5) to predict the answer $\hat{a}$ ($\hat{a} \in \mathcal{A}$). Proposed framework is depicted in Fig. 2.

3.1 Feature Extraction

A pretrained deep network based object detection model (Faster R-CNN, [27]) is used to identify the top-n_v salient regions from the input image I. The pretrained ResNet-101 [11] network is used to compute the visual feature of each region as an embedding $\mathbf{r} \in \mathbb{R}^{d_v}$. Thus, the input image I is represented as $\mathbf{rI} \in \mathbb{R}^{d_v \times n_v}$ by using n_v number of d_v dimensional ResNet-101 embeddings.

$$\mathbf{rI} = [\mathbf{r}_1, \ldots \mathbf{r}_{n_v}]; \mathbf{r} \in \mathbb{R}^{d_v} \tag{1}$$

The input natural language question q is first padded and trimmed to a length of n_w words. The word features are further extracted as pretrained GloVe embeddings [26] $\mathbf{eq} \in \mathbb{R}^{d_w}$ [26]. Thus, the question q is represented as $\mathbf{E_q} \in \mathbb{R}^{d_w \times n_w}$ by using n_w number of d_w dimensional embeddings.

$$\mathbf{E_q} = [\mathbf{eq}_1, \ldots \mathbf{eq}_{n_w}]; \mathbf{eq} \in \mathbb{R}^{d_w} \tag{2}$$

All feature embeddings in $\mathbf{rI}$ and $\mathbf{E_q}$ are projected to a common d dimensional space to obtain the respective initial feature embedding matrices as $\mathbf{rI}(0)$ and $\mathbf{E_q}(0)$.

$$\mathbf{rI(0)} = W_c^I \mathbf{rI} \tag{3}$$
$$\mathbf{Eq(0)} = W_c^Q \mathbf{E_q} \tag{4}$$

Here, $W_c^I \in \mathbb{R}^{d \times d_v}$ and $W_c^Q \in \mathbb{R}^{d \times d_w}$ are the transformation matrices. These representations are provided as input to the self- and co-attention modules.

3.2 Self-Attention

The self-attention (SA) mechanism is one of the key components of the proposed model. It is incorporated for both textual (question as collection of words) and visual (image as top-n_v salient regions) modalities. At the t^{th} ($t = 1, \ldots T$) block, the input to SA are $\mathbf{rI}(t-1)$ and $\mathbf{E_q}(t-1)$. Following [34], the SA uses *keys* and *queries*, both of dimension d_{KQ} and values of dimension d_{VS} respectively. The *Multi-Head Attention* [34] is incorporated to capture the attention from different aspects. For this, n_h parallel heads are added, where each head is considered to learn the relationships from different view (for image) and context (for question).

Let $\mathbf{E_M} = \{\mathbf{em}_1 \ldots \mathbf{em}_l\}$ be a matrix of feature embeddings, where $\mathbf{em} \in \mathbb{R}^{d_m}$ and $\mathbf{E_M} \in \mathbb{R}^{d_m \times l}$. For visual features, $\mathbf{E_M} = \mathbf{rI}(t-1)$, $l = n_v$ and $d_m = d$. Similarly, for question features, $\mathbf{E_M} = \mathbf{E_q}(t-1)$, $l = n_w$ and $d_m = d$.

The query ($Q_S^{(i)}$), key ($K_S^{(i)}$) and value ($V_S^{(i)}$) matrices for the i^{th} head can be respectively expressed as follows

$$Q_S^{(i)} = \left(W_i^{QS}\right)^\intercal \mathbf{E_M}, K_S^{(i)} = \left(W_i^{KS}\right)^\intercal \mathbf{E_M}, V_S^{(i)} = \left(W_i^{VS}\right)^\intercal \mathbf{E_M} \tag{5}$$

where, $W_i^{QS} \in \mathbb{R}^{d_m \times d_{KQ}}$, $W_i^{KS} \in \mathbb{R}^{d_m \times d_{KQ}}$ and $W_i^{VS} \in \mathbb{R}^{d_m \times d_{VS}}$ are transformation matrices. Using $\{Q_S^{(i)}, K_S^{(i)}, V_S^{(i)}\}$, the inner product of query is performed with all the keys and is divided by $\sqrt{d_k}$ for more stable gradients [34]. The *SoftMax* function is applied on the inner product to obtain the attention weights for question words and image salient regions. A scaled inner product based attention is computed for all the heads in the following manner.

$$\mathbf{H_i} = \left(V_S^{(i)}\right) \text{SoftMax}\left(\frac{{Q_S^{(i)}}^{\top} K_S^{(i)}}{\sqrt{d_K}}\right) \tag{6}$$

$$\mathbf{MH}(\mathbf{E_M}) = W_{mh}\mathbf{H} \tag{7}$$

Here, $W_{mh} \in \mathbb{R}^{d_m \times (n_h \times d_{VS})}$ is the transformation matrix. The output ($\text{MH}(\mathbf{E_M})$) of multi-head attention module is passed through fully connected feed forward layers with ReLU activation and dropout to prevent overfitting. Further, residual connections [11] followed by layer normalization are applied on top of fully connected layers for faster and more accurate training. The layer normalization is applied over the embedding dimension only. Finally, the self-attended embeddings of the input feature $\mathbf{E_M}$ are obtained as $\mathbf{SE_M} = \{\mathbf{sem}_1 \ldots \mathbf{sem}_l\}$ where $\mathbf{sem} \in \mathbb{R}^{d_m}$ and $\mathbf{SE_M} \in \mathbb{R}^{d_m \times l}$.

3.3 Co-Attention

For cross-modal interactions, the co-attention module intakes the representations of two modalities and generates attention in context of each other. To facilitate this, the self-attended embeddings $\widetilde{\mathbf{E_q}}(t-1)$ and $\widetilde{\mathbf{rI}}(t-1)$ are taken as input. For generating image attention in context of question words, keys and values are generated from self-attended intermediate question representation while the query is obtained from the image itself (following Eq. 6). Thus, the query $(Q_C^{(i)})$, key $(K_C^{(i)})$ and value $(V_C^{(i)})$ are respectively computed as follows.

$$Q_C^{(i)} = \left(W_i^{QC}\right)^{\intercal} \widetilde{\mathbf{E_q}}(t-1), K_C^{(i)} = \left(W_i^{KC}\right)^{\intercal} \widetilde{\mathbf{rI}}(t-1), V_C^{(i)} = \left(W_i^{VC}\right)^{\intercal} \widetilde{\mathbf{E_q}}(t-1) \tag{8}$$

Here, $W_i^{QC} \in \mathbb{R}^{d_m \times d_{KQ}}$, $W_i^{KC} \in \mathbb{R}^{d_m \times d_{KQ}}$ and $W_i^{VC} \in \mathbb{R}^{d_m \times d_{KV}}$ are transformation matrices. Similarly, for cross-modal question attention, the query is obtained from self-attended question embeddings. While the keys and values are obtained from self-attended image embeddings. These queries, keys and values are similarly processed following Eqs. 6 and 7 to obtain the multi-head attention. This is fed to fully connected layers with ReLU, dropout, skip connections and layer normalization. The output of this network provides the final output of the co-attention module.

3.4 Cascading and Fusion

A single SCA block comprising of *self-attention* (intra-modality interaction) and *co-attention* (inter-modality interaction) generates an enriched representation $(\mathbf{rI}(t), \mathbf{E_q}(t))$ of its input visual and textual features.

First block takes $\mathbf{rI}(\mathbf{0})$ and $\mathbf{Eq}(\mathbf{0})$ as input to respective SA module. Output of each visual and textual self attended features serves as the input to cross attention modules (text-to-image, image-to-text) for further refining the contextual information. This flow of feature refinement is cascaded for multiple SCA block to T steps. Let $\mathbf{rI}(T) \in \mathbb{R}^{d \times n_v}$ and $\mathbf{E_q}(T) \in \mathbb{R}^{d \times n_w}$ be the respective visual and question representations obtained from the final (T^{th}) SCA block.

The feature representations are obtained by averaging the attended embeddings of two modalities. So, the final visual embedding, say $\mathbf{I}_f$ is obtained as follows.

$$\mathbf{I}_f = \frac{1}{k} \sum_{j=1}^{n_v} \mathbf{rI}(T)[:, j] \tag{9}$$

Similarly, the question encoding, say $\mathbf{Q}_f$ is evaluated in the following manner.

$$\mathbf{Q}_f = \frac{1}{n_w} \sum_{j=1}^{n_w} \mathbf{E_q}(T)[:, j] \tag{10}$$

The unified multi-modal representation, say $\mathbf{F} \in \mathbb{R}^d$ is obtained by fusing $\mathbf{I}_f$ and $\mathbf{Q}_f$ through element-wise multiplication ($\odot$).

$$\mathbf{F} = \mathbf{I}_f \odot \mathbf{Q}_f \tag{11}$$

The fused embedding $\mathbf{F}$ is fed to a fully connected network for answer prediction.

3.5 Answer Prediction

The fused embedding $\mathbf{F}$ is fed to fully connected network with single hidden layer of dimension d_{hp}. The number of labels at the output layer is n_c ($n_c =\mid \mathcal{A} \mid$). The output answer vector, say $\hat{\mathbf{a}}$ is predicted as follows.

$$\hat{\mathbf{a}} = \text{FCNet}\left(\mathbf{F}; d_{hp}; n_c\right) \tag{12}$$

3.6 Model Learning

Let the respective ground truth and predicted answer be a and $\hat{a}$ ($a, \hat{a} \in \mathcal{A}$) for input image I and question Q. This model uses cross-entropy loss for answer prediction and is defined as

$$\mathcal{L}_c = -\sum_{j=1}^{n_c} a[j] log(\hat{a}[j]) \tag{13}$$

The combined set of parameters for proposed model includes the ones for feature extraction, block of dense attention and fusion mechanism.

4 Experiment Design

This section discusses the datasets used to benchmark the proposed model, the three evaluation metrics and the necessary implementation details.

4.1 Dataset

The proposed model is evaluated through experiments performed on the datasets VQA2.0 [10], TDIUC [15] and GQA [13]. The VQA2.0 [10] dataset is one of the most commonly used for the VQA task. The dataset is divided into *train*, *validation* and *test* sets with 443757, 214354 and 447793 image, question and answer triplets respectively. The Task-Directed Image Understanding Challenge (TDIUC) [15] is another large VQA dataset of real images. Questions are categorized into 12 types.

Total 1.6 million question, image and answer triplets are split into *train* and *validation* sets. The *train* set consists of 1.1 million triplets and 0.5 million triplets are in the validation split. To deal with language prior issues, TDIUC consists of a special category 'Absurd', where an input question is not related to the visual content of a given image. GQA [13] is the largest VQA dataset consisting of compositional questions based on real-world images. "Balanced split" set of the GQA dataset consists of $1M$ questions and ensures a better equitable distribution of answers.

4.2 Evaluation Metrics

For evaluation of the TDIUC dataset, *Arithmetic-Mean Per Type (AMPT)* and *Harmonic-Mean Per Type (HMPT)* are proposed in [15] as fair evaluation metrics along with *Overall Accuracy.* The AMPT is the average of question category-wise accuracies with uniform weight to each category. On the other hand, HMPT measures the ability of the model to have a high score across all question types.

The VQA2.0 dataset evaluation is performed using the following metric defined in [2].

$$\mathbf{Accuracy}(\hat{\mathbf{a}}) = min\left\{\frac{\mathbf{\#humans\ that\ said\ \hat{a}}}{\mathbf{3}}, \mathbf{1}\right\} \tag{14}$$

Each question in the VQA2.0 dataset was answered by 10 annotators. The above evaluation metric considers a predicted answer correct if it matches the answers given by at least 3 annotators.

For GQA dataset, standard accuracy is used, where 1 point is given if the predicted answer $\hat{a}$ matches the ground truth answer a and 0 otherwise. The final results are described as average over all questions in the dataset.

5 Results and Discussion

5.1 Quantitative Results

Overall Performance & Category-Wise Performance Comparison on TDIUC Dataset – Table 1 and 3 present the respective class-wise and overall performance for the TDIUC dataset. In terms of the overall accuracy, Arithmetic-MPT (AMPT) and Harmonic-MPT (HMPT) measures, the proposed model CSCA exhibits better performance compared to most of the baseline methods. Also, in terms of class-wise accuracy, CSCA leads in all except one class. A significant relative gain of 12.6% is observed compared to the next best performing model for the *'Counting'* category of questions. Table 4 presents the results for different models trained *'Without Absurd'* category of questions. It is observed that CSCA performs better than the existing ones for all three defined metrics.

Table 1. Category-wise comparison of CSCA with previous state-of-the-art methods on the TDIUC dataset

Question Type	SAN [39]	RAU [15]	MCB [9]	QTA [29]	BAN [16]	CSCA
Scene Recognition	92.3	93.96	93.06	93.80	93.1	**94.48**
Sport Recognition	95.5	93.47	92.77	95.55	95.7	**95.85**
Color Attributes	60.9	66.86	68.54	60.16	67.5	**75.51**
Other Attributes	46.2	56.49	56.72	54.36	53.2	**60.89**
Activity Recognition	51.40	51.60	52.35	60.10	54.0	**61.00**
Positional Reasoning	27.9	35.26	35.40	34.71	27.9	**42.14**
Object Recognition	87.50	86.11	85.54	86.98	87.5	**89.11**
Absurd	93.4	96.08	84.82	**100.0**	94.47	97.28
Utility & Affordance	26.3	31.58	35.09	31.48	24.0	**40.35**
Object Presence	92.4	94.38	93.64	94.55	95.1	**96.34**
Counting	52.1	48.43	51.01	53.25	53.9	**60.70**
Sentiment Und.	53.6	60.09	66.25	64.38	58.7	**67.19**
Overall Accuracy	82.0	84.26	81.86	85.03	85.5	**88.12**
Harmonic Mean	53.7	59.00	60.47	60.08	54.9	**67.05**
Arithmetic Mean	65.0	67.81	67.90	69.11	67.4	**73.34**

Overall Performance and Category-Wise Performance Comparison on VQA2.0 Dataset – Table 2 demonstrates the results on test-dev and test-std splits of the VQA2.0 dataset. Performance of the proposed model CSCA is comparable with that of the best among the existing methods. The models LXMERT [33], ViLBERT [20] are pre-trained for multiple vision and language based tasks and are fine-tuned for VQA. Here, CSCA has obtained 67.36% accuracy on the validation set. This is around 1% improvement over the best performance among the existing methods.

Overall Performance Comparison on GQA Dataset – Table 6 presents the results for the "*balanced split*" of the GQA dataset. For a fair comparison and as per the availability of results from respective papers, the results are reported in terms of overall accuracy. The GQA dataset emphasizes the need for multi-hop reasoning for evaluating the reasoning abilities of model proposals.

It is observed that the competitive methods either rely on graph-based structures or incorporate information from external knowledge base to enhance visual reasoning capability [12,28,35,44–46]. However, CSCA achieves a performance without relying on graph-based structures or external knowledge. Specifically, CSCA demonstrates a better performance (an improvement of 0.8%) compared to NSM [12] (best among all).

Table 2. Model performance on VQA 2.0 dataset: Validation, Test-Dev & Test-Std splits. CSCA is compared with several state-of-the-art methods including *Fusion based*, *Visual Attention*, *Dense Attention* based methods

Methods	Val	Test-Dev				Test-Std
	Overall	Yes/No	Number	Other	Overall	Overall
MCB [6]	59.14	78.46	38.28	57.80	62.27	53.36
MLB [17]	62.98	83.58	44.92	56.34	66.27	66.62
MUTAN [3]	62.71	82.88	44.54	56.50	66.01	66.38
MFH [43]	62.98	84.27	49.56	59.89	68.76	–
BLOCK [4]	64.91	83.14	51.62	58.97	68.09	68.41
SAN [39]	61.70	78.40	40.71	54.36	61.70	–
BTUP [1]	63.20	81.82	44.21	56.05	65.32	65.67
BAN [16]	65.81	82.16	45.45	55.70	64.30	–
v-VRANet [40]	–	83.31	45.51	58.41	67.20	67.34
ALMA [19]	–	84.62	47.08	58.24	68.12	66.62
ODA [47]	64.23	83.73	47.02	56.57	66.67	66.87
BAN2-CTI [5]	66.00	–	–	–	–	67.4
CRANet [25]	–	83.31	45.51	58.41	67.20	67.34
CoR [36]	65.14	84.98	47.19	58.64	68.19	68.59
DFAF [7]	66.66	86.09	53.32	60.49	70.22	70.34
MLIN [8]	66.53	85.96	52.93	60.40	70.18	70.28
LXMERT [33]	–	–	–	–	–	**72.5**
ViLBERT [20]	–				70.55	70.92
VSDC [35]	65.39	83.79	48.16	59.31	68.55	68.67
RSL [37]	66.77	86.94	51.37	61.09	70.64	71.06
VQA-BC [18]	61.74	–	–	–	–	–
BCO [14]	63.80	–	–	–	–	–
EDC [23]	–	83.98	48.15	58.74	67.94	68.14
CSCA	**67.36**	86.57	**53.58**	**61.06**	**70.72**	71.04

Table 3. Comparing *Overall Accuracy* for TDIUC dataset

Model	Overall Accuracy	Arithmetic Mean
BTUP [1]	82.91	68.82
QCG [24]	82.05	65.67
CTI [5]	87.00	72.5
DFAF [7]	85.55	**NA**
RAMEN [31]	86.86	72.52
MLIN [8]	87.60	NA
CSCA	**88.12**	**73.34**

Table 4. Performance of CSCA on TDIUC data (except Absurd category samples) trained without 'Absurd' Category samples

Metrics	Overall Accuracy	Arithmetic MPT	Harmonic MPT
MCB [9]	78.06	66.07	55.43
QTA [29]	80.95	66.88	58.82
BAN [16]	81.9	64.6	52.8
CTI [5]	85.0	70.6	63.8
CSCA	**85.30**	**71.21**	**65.40**

5.2 Ablation Analysis

The proposed model performs self-attention on the two modalities to obtain intra-modality correlated features. Then the co-attention module uses respective representations of the two modalities to obtain cross-modality correlated features by performing attention for one modality in the context of another. In this ablation analysis, we examine the impact of individual attention module in various combinations to understand their importance. We also analyze the set of correct predictions obtained in these settings.

Table 5. Evaluating model performance on VQA2.0, TDIUC & GQA dataset to investigate the effect of *different basic attention modules* of the proposed model

SA	CA	VQA2.0		TDIUC		GQA	
		Overall Accuracy	Parameter (in Millions)	Overall Accuracy	Parameter	Overall (in Millions)	Parameter (in Millions)
✗	✗	55.80	15	69.18	7	49.11	4.47
✗	✓	59.69	22	70.46	21	53.58	20.08
✓	✗	64.13	25	87.42	25	57.82	23.24
✓	✓	**67.36**	**42**	**88.12**	**36**	**63.60**	**35.85**

Effect of Different Modules in SCA Block – In the Table-5 we present the results of ablation analysis experiments in terms of performance and complexity. The complexity is expressed in terms of the number of model parameters.

The first row of the table shows the model performance when neither of the attention is incorporated. The features for both modalities are fused directly via element-wise multiplication without applying self- or co-attention. Second row shows the performance when *only self-attention* (SA only) is incorporated on both modalities and answer prediction is based on the fused embedding of the self-attended representations of the individual modalities. Here, the fused representation is obtained via element-wise multiplication. Third row shows the results when *only co-attention* (CA only) is incorporated on image and question in the context of the other. The last row shows the results from the proposed model that comprises of both *self-attention* and *co-attention* in cascade (SCA).

Table 6. Comparing *Overall Accuracy* of CSCA for GQA

Model	Overall Accuracy
NSM [12]	63.17
XNM [28]	62.04
OCCAM [35]	63.10
XKI [44]	62.38
QAA [45]	63.07
DDTN [46]	58.54
CSCA	**63.60**

As per expectation, the model without any attention mechanism provides the lowest performance (first row). The "SA only" model provides lower performance as it lacks the interaction of two modalities and learns a comparatively poor representation (second row). Co-attention is the crucial component for multi-modality that is found to perform better than *self-attention*. In terms

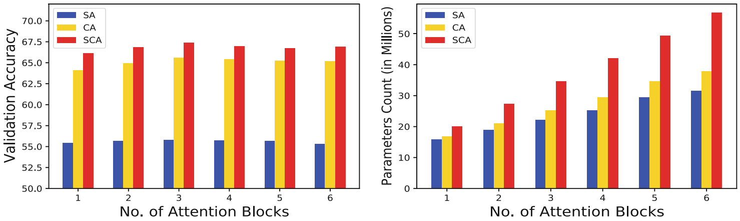


Fig. 3. Number of attention blocks incorporated. [Left] Validation accuracy for VQA2.0 *'val'* split with respect to attention blocks. [Right] Parameter counts with respect to attention blocks

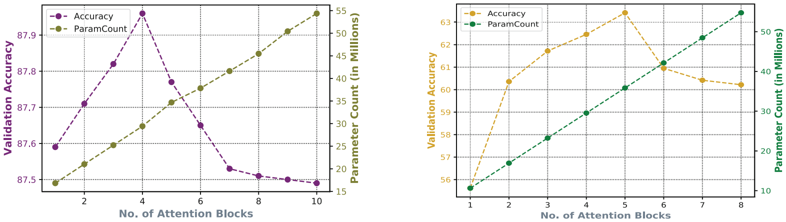


Fig. 4. The effect of number of attention blocks incorporated on the *validation* accuracy and *parameter count*[Left] For TDIUC*'val'* [Right] For GQA dataset

of computational complexity, a simple fusion-based model uses the least number of parameters, while the proposed model (SCA) requires the highest number of parameters. However, the performance improvement, especially for VQA2.0 dataset, overcomes the complexity issue. We observe that the change in model performance is similar for all three datasets in this analysis.

Effect of Number of SCA Blocks – It is difficult for a model to grasp all relevant information through a representation in one pass. Thus, attention blocks in cascade extract the fine-grained information and pass it on to the next one for further refinement. We perform experiments to identify the optimal number of blocks in the cascade. The effect of different independent attention mechanisms (SA only, CA only, SCA) for answer prediction is also analyzed. In Fig. 3 (left), the overall performance for the validation split of the VQA2.0 dataset is given concerning varying numbers of blocks. Figure 3 (right) shows the parameter counts with respect to the number of blocks. As per expectation, we can observe that the models perform poorly with single attention blocks (SA only, CA only, SCA). However, the performance is observed to rise only up to four blocks. Increasing the number of blocks beyond four does not lead to any further performance improvement. Adding more blocks also increases the number of model parameters (Fig. 3). Furthermore, one can observe that only the CA module can perform better than using only the SA module. This is as per the expectation. Similarly, Fig. 4 (left) shows that the model performance keeps improving until the fourth SCA block for the TDIUC dataset. The model performance starts deteriorating with a further increase in blocks. For the GQA dataset, Fig. 4 demonstrates a similar observation. It is worth noting that the performance steadily improves until the fifth block of the SCA. However, a drop of approximately 2% in performance is observed on the sixth block. After this point, the performance continues to degrade.

6 Conclusion

This work proposes a dense attention mechanism-based VQA model. Dense attention is incorporated by exploiting both self-attention and co-attention. The self-attention mechanism helps in obtaining improved representation within a single modality. With self-attention, a salient region (in the case of image) interacts with every other region. The final representation inherits the contextual information for all regions. Similarly, for the input questions, self-attention provides the representation of every single word that captures the contextual information for other words as well. The proposed model also exploits the cross-modal interaction of two modalities which is further strengthened by self-attention of two modalities. Attention blocks are cascaded multiple times to facilitate refined cues of visual and textual features. The model's capability is justified by detailed experiments and analysis performed on the three benchmark VQA datasets.

References

1. Anderson, P., et al.: Bottom-up and top-down attention for image captioning and visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6077–6086 (2018)
2. Antol, S., et al.: VQA: visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2425–2433 (2015)
3. Ben-Younes, H., Cadene, R., Cord, M., Thome, N.: MUTAN: multimodal tucker fusion for visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2612–2620 (2017)
4. Ben-Younes, H., Cadene, R., Thome, N., Cord, M.: BLOCK: bilinear superdiagonal fusion for visual question answering and visual relationship detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 8102–8109 (2019)
5. Do, T., Do, T.T., Tran, H., Tjiputra, E., Tran, Q.D.: Compact trilinear interaction for visual question answering. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 392–401 (2019)
6. Fukui, A., Park, D.H., Yang, D., Rohrbach, A., Darrell, T., Rohrbach, M.: Multimodal compact bilinear pooling for visual question answering and visual grounding. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 457–468. Austin, Texas, November 2016
7. Gao, P., et al.: Dynamic fusion with intra-and inter-modality attention flow for visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6639–6648 (2019)
8. Gao, P., You, H., Zhang, Z., Wang, X., Li, H.: Multi-modality latent interaction network for visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5825–5835 (2019)

9. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact bilinear pooling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 317–326 (2016)
10. Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., Parikh, D.: Making the V in VQA matter: elevating the role of image understanding in visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6904–6913 (2017)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
12. Hudson, D., Manning, C.D.: Learning by abstraction: the neural state machine. In: Advances in Neural Information Processing Systems, vol. 32 (2019)
13. Hudson, D.A., Manning, C.D.: Gqa: a new dataset for real-world visual reasoning and compositional question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6700–6709 (2019)
14. Jha, A., Patro, B., Van Gool, L., Tuytelaars, T.: Barlow constrained optimization for visual question answering. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 1084–1093 (2023)
15. Kafle, K., Kanan, C.: An analysis of visual question answering algorithms. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1965–1973 (2017)
16. Kim, J.H., Jun, J., Zhang, B.T.: Bilinear attention networks. In: Advances in Neural Information Processing Systems, pp. 1564–1574 (2018)
17. Kim, J.H., On, K.W., Lim, W., Kim, J., Ha, J.W., Zhang, B.T.: Hadamard product for low-rank bilinear pooling. arXiv preprint arXiv:1610.04325 (2016)
18. Lao, M., Guo, Y., Chen, W., Pu, N., Lew, M.S.: Vqa-bc: robust visual question answering via bidirectional chaining. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4833–4837. IEEE (2022)
19. Liu, Y., Zhang, X., Huang, F., Cheng, L., Li, Z.: Adversarial learning with multimodal attention for visual question answering. IEEE Trans. Neural Netw. Learn. Syst. (2020)
20. Lu, J., Batra, D., Parikh, D., Lee, S.: Vilbert: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. arXiv preprint arXiv:1908.02265 (2019)
21. Mishra, A., Anand, A., Guha, P.: CQ-VQA: visual question answering on categorized questions. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE (2020)
22. Mishra, A., Anand, A., Guha, P.: Multi-stage attention based visual question answering. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 9407–9414. IEEE (2021)
23. Mohamud, S.A.M., Jalali, A., Lee, M.: Encoder-decoder cycle for visual question answering based on perception-action cycle. Pattern Recogn. **144**, 109848 (2023)
24. Norcliffe-Brown, W., Vafeias, S., Parisot, S.: Learning conditioned graph structures for interpretable visual question answering. In: Advances in Neural Information Processing Systems, pp. 8334–8343 (2018)
25. Peng, L., Yang, Y., Wang, Z., Wu, X., Huang, Z.: Cra-net: composed relation attention network for visual question answering. In: Proceedings of the 27th ACM International Conference on Multimedia, pp. 1202–1210 (2019)
26. Pennington, J., Socher, R., Manning, C.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)

27. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
28. Shi, J., Zhang, H., Li, J.: Explainable and explicit visual reasoning over scene graphs. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8376–8384 (2019)
29. Shi, Y., Furlanello, T., Zha, S., Anandkumar, A.: Question type guided attention in visual question answering. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 151–166 (2018)
30. Shih, K.J., Singh, S., Hoiem, D.: Where to look: focus regions for visual question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4613–4621 (2016)
31. Shrestha, R., Kafle, K., Kanan, C.: Answer them all! Toward universal visual question answering models. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 10472–10481 (2019)
32. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
33. Tan, H., Bansal, M.: Lxmert: learning cross-modality encoder representations from transformers. arXiv preprint arXiv:1908.07490 (2019)
34. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
35. Wang, Z., et al.: Interpretable visual reasoning via induced symbolic space. In: Proceedings of the CVF International Conference on Computer Vision, pp. 1878–1887 (2021)
36. Wu, C., Liu, J., Wang, X., Dong, X.: Chain of reasoning for visual question answering. Adv. Neural. Inf. Process. Syst. **31**, 275–285 (2018)
37. Xiao, X., Zhang, C., Xiang, S., Pan, C.: Reinforcement stacked learning with semantic-associated attention for visual question answering. In: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4170–4174. IEEE (2021)
38. Xu, H., Saenko, K.: Ask, attend and answer: exploring question-guided spatial attention for visual question answering. In: European Conference on Computer Vision, pp. 451–466. Springer (2016)
39. Yang, Z., He, X., Gao, J., Deng, L., Smola, A.: Stacked attention networks for image question answering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 21–29 (2016)
40. Yu, J., et al.: Reasoning on the relation: enhancing visual representation for visual question answering and cross-modal retrieval. IEEE Trans. Multimedia **22**(12), 3196–3209 (2020)
41. Yu, Z., Yu, J., Cui, Y., Tao, D., Tian, Q.: Deep modular co-attention networks for visual question answering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6281–6290 (2019)
42. Yu, Z., Yu, J., Fan, J., Tao, D.: Multi-modal factorized bilinear pooling with co-attention learning for visual question answering. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1821–1830 (2017)
43. Yu, Z., Yu, J., Xiang, C., Fan, J., Tao, D.: Beyond bilinear: generalized multimodal factorized high-order pooling for visual question answering. IEEE Trans. Neural Netw. Learn. Syst. **99**, 1–13 (2018)
44. Zhang, Y., Jiang, M., Zhao, Q.: Explicit knowledge incorporation for visual reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1356–1365 (2021)

45. Zhang, Y., Jiang, M., Zhao, Q.: Query and attention augmentation for knowledge-based explainable reasoning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15576–15585 (2022)
46. Zhu, L., Peng, L., Zhou, W., Yang, J.: Dual-decoder transformer network for answer grounding in VQA. Pattern Recogn. Lett. **171**, 53–60 (2023)
47. Zhu, X., Mao, Z., Chen, Z., Li, Y., Wang, Z., Wang, B.: Object-difference drived graph convolutional networks for visual question answering. Multimedia Tools Appl. **80**(11), 16247–16265 (2021)

Facet-Aware Multimodal Summarization via Cross-Modal Alignment

Yu Weng[1,2], Xuming Ye[1,2], Tianjiao Xing[1,2], Zheng Liu[1,2](✉), Chaomurilige[1,2](✉), and Xuan Liu[1,2]

[1] Minzu University of China, Beijing 100081, China
{wengyu,xumingye,22302012,liuzheng,chaomurilige,liuxuan}@muc.edu.cn
[2] Key Laboratory of Ethnic Language Intelligent Analysis and Security Governance of MOE, Minzu University of China, Beijing, China

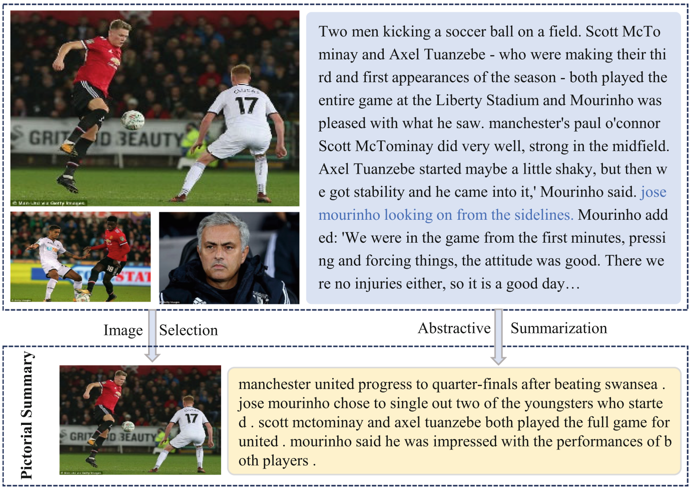
Abstract. Multimodal generative models have demonstrated promising capabilities for bridging the semantic gap between visual and textual modalities, especially in the context of multimodal summarization. Most of the existing methods align the visual and textual information by self-attention mechanism. However, those approaches will cause imbalances or discrepancies between different modalities when processing such text-heavy tasks. To address this challenge, our method introduces an innovative multimodal summarization method. We first propose a novel text-caption alignment mechanism, which considers the semantic association across modalities while maintaining the semantic information. Then, we introduce a document segmentation module with a salient information retrieval strategy to integrate the inherent semantic information across facet-aware semantic blocks, obtaining a more informative and readable textual output. Additionally, we leverage the generated text summary to optimize image selection, enhancing the consistency of the multimodal output. By incorporating the textual information in the image selection process, our method selects more relevant and representative visual content, further enhancing the quality of the multimodal summarization. Experimental results illustrate that our method outperforms existing methods by utilizing visual information to generate a better text-image summary and achieves higher ROUGE scores.

Keywords: Multimedia analysis · Document understanding · Semantic technology · Summarization

1 Introduction

Multimodal summarization aims to generate a concise and informative summary by jointly analyzing and fusing information from multiple modalities, such as text and images [6,10,16]. In contrast to text summarization, which relies on a single data type, multimodal summarization requires advanced neural models to encode multimodal data, capture inter-modal interactions, and generate an

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 37–52, 2025.
https://doi.org/10.1007/978-3-031-78495-8_3



Two men kicking a soccer ball on a field. Scott McTo
minay and Axel Tuanzebe - who were making their thi
rd and first appearances of the season - both played the
entire game at the Liberty Stadium and Mourinho was
pleased with what he saw. manchester's paul o'connor
Scott McTominay did very well, strong in the midfield.
Axel Tuanzebe started maybe a little shaky, but then w
e got stability and he came into it,' Mourinho said. jose
mourinho looking on from the sidelines. Mourinho add
ed: 'We were in the game from the first minutes, pressi
ng and forcing things, the attitude was good. There we
re no injuries either, so it is a good day…

manchester united progress to quarter-finals after beating swansea .
jose mourinho chose to single out two of the youngsters who starte
d . scott mctominay and axel tuanzebe both played the full game for
united . mourinho said he was impressed with the performances of b
oth players .

Fig. 1. The illustration of our proposed task Multimodal Summarization with Multimodal Output (MSMO). Image and text assistance can generate a richer summary for easy understanding.

integrated text summary. This paper focuses on the Text-Image Summarization (TIS) task [6,10,27], which considers both image and text modalities (Fig. 1).

Current multimodal summarization methods typically involve three stages: 1) feature extraction, 2) modality fusion, and 3) summary generation [2,17,24]. Most approaches separately extract textual and visual features using techniques such as Convolutional Neural Networks (CNN) [1] for visual encoding and Recurrent Neural Networks (RNN) [20] for natural language processing and generation [11,14,23,25]. Subsequently, cross-modal attention mechanisms are employed to fuse the multimodal semantic information [24]. However, traditional multimodal summarization methods often struggle to effectively leverage visual information to improve the text summary, and in some cases, the visual information may even hinder the summarization performance.

Previous multimodal works transform multiple modalities into a single modality to improve the performance of multimodal tasks [5,8,21,22]. For example, image captioning [8,21] can automatically generate descriptive text for a given image called image caption, providing a comprehensive understanding of its content. Inspired by the image caption (e.g. CLIP [18] and BLIP [12]), we observed that it could naturally transform the visual modality into text modality, thereby translating the multimodal summarization task into the text summarization task. However, upon transforming the multimodal summarization into the text summarization through image captioning, we face the problem that a single document often encompasses multiple themes or facets [19,26]. Also, we found that the summary generated from text augmented with image captions contained irrelevant information, necessitating the filtering of irrelevant content from the text to enhance summary quality.

Based on our observations, we propose a novel multimodal summarization method, involving four modules: Text-Caption Alignment, Facet-Aware Document Segmentation, Salient Information Retrieval strategy, and Image Selection. Extensive experiments on the existing dataset demonstrate that our method not only fully exploits visual information to generate a more comprehensive multimodal summary, but also ensures the reduction of irrelevant information without the removal of useful data and generates an accurate summary. Our contributions are as follows:

- First, we introduce a strategy for Text-Caption Alignment based on semantic localization, which can integrate visual information into textual content, avoiding issues such as an illogical summary caused by inserting the captions directly and enabling the generation of a more comprehensive and fluent text summary.
- Second, we propose a sentence-level information filtering technique, capable of filtering extraneous information within the text. Which enhances the informativeness and precision of the generated summary, ultimately achieving state-of-the-art performance as measured by the ROUGE metric.
- Third, we propose a straightforward yet effective Image Selection strategy. By projecting the generated summary and images into a joint textual semantic space, we can select images with a higher degree of relevance to the generated text summary.

2 Related Work

Multimodal summarization processes data from diverse modalities to generate more concise information, improving the quality of the summary. For multimodal output with multimodal input [24,27,28], Zhu et al. [27] constructed a large-scale dataset for TIS task, which takes the images and text as input and outputs a pictorial summary. They also proposed a multimodal attention model to jointly generate text and select the most relevant image from the multimodal input. Based on previous research, Zhu et al. [28] proposed a multimodal objective function with the guidance of multimodal reference to use the loss from the summary generation and the image selection. Zhang et al. [24] introduced a unified framework for multimodal summarization, called UniMS, which integrates extractive and abstractive objectives, as well as selecting the image output.

However, exploiting the multimodal information is difficult because of the semantic bias during multimodal fusion. To address the challenge, some works align the multimodal information into a shared space. For instance, Ding et al. [5] introduce a contrastive loss to align the image and audio representations before fusing them through cross-modal attention. Pereira et al. [22] focuses on capturing visual sentiment information through facial expressions in text and selectively matching and fusing with the target aspect in textual modality. Wu et al. [21] proposed a method to insert external knowledge into the CNN-RNN for visual question answering, transforming the images into image captions. La et al. [8] utilized image captioning to extract the correlation between different modalities to solve multimodal tasks. Inspired by converting the multimodal task into the unimodal task, we transform multimodal summarization into text summarization, which can reduce the complexity of data fusion and improve the performance of multimodal summarization.

3 Proposed Method

3.1 Method Overview

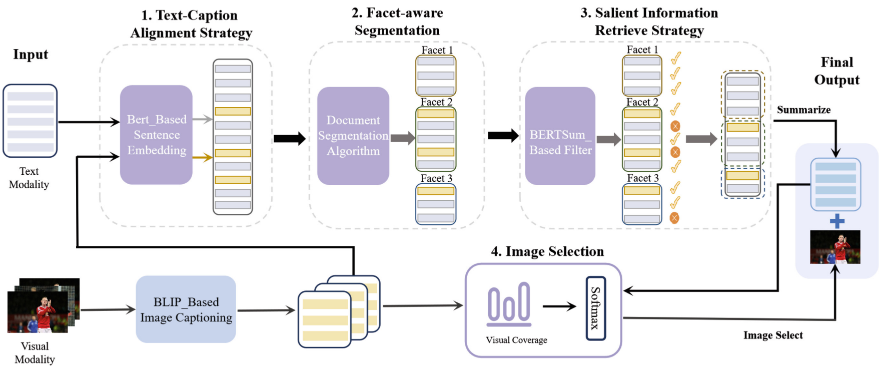

1. Text-Caption Alignment Strategy
2. Facet-aware Segmentation
3. Salient Information Retrieve Strategy
Input
Text Modality
Bert_Based Sentence Embedding
Document Segmentation Algorithm
Facet 1
Facet 2
Facet 3
BERTSum_ Based Filter
Final Output
Summarize
4. Image Selection
Visual Modality
BLIP_Based Image Captioning
Visual Coverage
Softmax
Image Select

Fig. 2. Overview of our work. Specifically, we mainly divide it into four parts: (1) After generating image captions, the image is inserted into the text based on semantic similarity (2) Then, the document is segmented using a document segmentation algorithm, which is called the Facet-Aware Segmentation. (3) By filtering out the irrelevant sentences in each block, we select the most relevant sentences for summarization. (4) We select the image with the highest semantic similarity between the image description and the text summary as the image summary.

In our paper, we propose a novel multimodal summarization approach, where the input is a sequence of text and a set of images; the output contains a text and an image. As shown in Fig. 2, our method mainly consists of four modules: (1) **Text-Caption Alignment**, (2) **Facet-Aware Document Segmentation**, (3) **Salient Information Retrieval Strategy**, and (4) **Image Selection**.

Image captions are the textual representation of visual knowledge obtained from images and inserted into the optimal position in the original text document based on semantic similarity between images and text, as detailed in Sect. 3.2. As described in Sect. 3.3, we utilize a document segmentation strategy, which divides the document into facet-aware semantic blocks to keep semantic relevance between sentences within each block. Then, we filter out the irrelevant sentences from each block according to a hypermeter θ and generate the text summary from the filtered sentences, as described in Sect. 3.4. Finally, for image selection, we project the generated text summary and all image captions into the same textual semantic space and select the most relevant image compared with the text summary, as described in Sect. 3.5.

3.2 Text-Caption Alignment

Given the multimodal document $\{T, I\}$, we first transform multiple image modal $I = \{I_1,I_2,\ldots,I_j,\ldots,I_m\}$ into text modality $\tilde{I} = \{\tilde{I_1}, \tilde{I_2},\ldots, \tilde{I_j},\ldots, \tilde{I_m}\}$ by a pre-trained BLIP [12] model, where $\tilde{I_j}$ represents the corresponding image caption of the j-th image. Basically, it reduces the modality-bias problem in such text-heavy multimodal summarization task and thereby improve the performance of multimodal summarization.

We sequentially process each image caption generated by the BLIP model $\{\tilde{I_1}, \tilde{I_2},\ldots, \tilde{I_i},\ldots, \tilde{I_m}\}$, and calculate the semantic similarity with all paragraphs in the document $\{p_1, p_2,\ldots, p_j,\ldots, p_n\}$. In detail, we first obtain embedding vectors $\tilde{I_i}$ and p_j of the image caption and each paragraph in the text document through a pre-trained model. Then, the similarity of both sides of the potential insert point is measured by cosine similarity. The equation is as follows:

$$\alpha(\tilde{I_i}, p_j) = \frac{\tilde{I_i} \cdot p_j}{\left|\left|\tilde{I_i}\right|\right| ||p_j||} \tag{1}$$

where $\tilde{I_i}$ and p_j represent the i-th image caption and the j-th paragraph respectively. The purpose of calculating cosine similarity is to measure the directional similarity between two vectors, with the value range spanning from -1 to 1. The cosine similarity value closer to 1 indicates higher semantic similarity between the two sentences.

Upon obtaining the similarity scores between image descriptions and textual paragraphs, we average similarity scores on both sides of the potential insert point, serving as the matching score for that insertion point. Finally, the image captions are inserted into the text based on the maximum matching score.

3.3 Facet-Aware Document Segmentation

By employing a document segmentation algorithm to partition the document into k facet-aware semantic blocks, each block comprises distinct facets. Our approach is grounded in the assumption proposed by Skorokhod'ko [13] that when adjacent sentences exhibit semantic similarity, they converge on the same aspect. Building upon this aforementioned assumption, we predefine potential segmentation points $(g_1, g_2, \ldots, g_{n-1})$ at the breakpoints between every two sentences, where n is the number of sentences.

To determine whether a potential segmentation point is a true breakpoint, we calculate a score for each potential segmentation point based on the semantic similarity of the sentences surrounding it. Specifically, we first compute the features of the w sentences to the left and right of each segmentation point, and then calculate their similarity as the score for that segmentation point. As shown in Eq. 2, g_i^l and g_i^r respectively denote the left and right features at the segmentation point g_i. The similarity of the segmentation point g_i is computed using cosine similarity $sim_i = \frac{g_i^l \cdot g_i^r}{\|g_i^l\| \|g_i^r\|}$. It's noted that the default value for w is set to 2.

$$g_i^l = \frac{1}{w} \sum_{j=i-w+1}^{i} x_j, g_i^r = \frac{1}{w} \sum_{j=i+1}^{i+w} x_j \tag{2}$$

Inspired by TextTiling [7], we transform the similarity score into a depth score through the following equation:

$$\begin{aligned} d_i &= \mathbf{max}\{(sim_{i-1} - sim_i), 0\} \\ &+ \mathbf{max}\{(sim_{i+1} - sim_i), 0\} \end{aligned} \tag{3}$$

When the similarity of the potential segmentation point is the local minimum value, the facets in the left and right blocks are different, and the corresponding depth score is high. Thus, if d_i surpasses a threshold δ determined by the mean and standard deviation of the depth score sequence, the original predefined segmentation point g_i is chosen. The threshold δ controls the degree of document segmentation, where a higher δ results in more sentences per segmented block. Finally, we obtain the facet-aware blocks $(b_1, b_2, ..., b_k)$.

3.4 Salient Information Retrieval Strategy

Following the segmentation of the document, sentences within each block are related to a distinct facet. In this section, we detail how to filter out the irrelevant information within each block based on the importance of the block. At first, we calculate the semantic vector representation $(t_1, \ldots, t_i, \ldots, t_k)$ for each block as the Eq. 4:

$$t_i = \frac{1}{|b_i|} \sum_{x_i \in b_i} x_j \tag{4}$$

where $|b_i|$ represents the number of sentences in the segmented block b_i. Following Zheng et al. [26], we employ directed centrality to score each block, as shown in Eq. 5. Specifically, we obtain the centrality score $\rho(b_i)$ for each block by computing the sum of pair-wise dot product with other blocks. Noted that a high centrality score for a block indicates that the block is highly relevant and connected to other blocks within the document.

$$\rho(b_i) = \sum_{j \neq i}^{k} t_i \cdot t_j \tag{5}$$

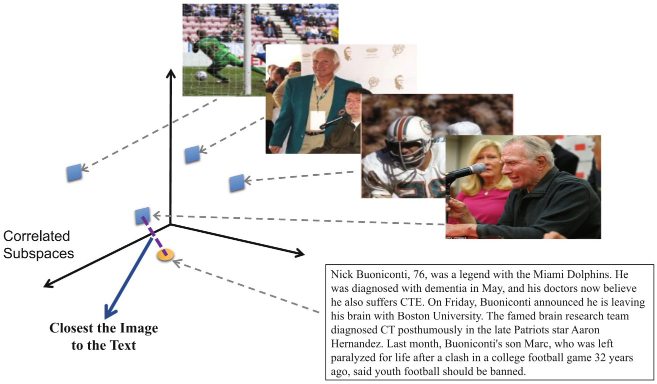
To filter out irrelevant sentences, we calculate the proportion of sentences to be retain within each block, denoted as η_i:

$$\eta_i = \frac{\rho(b_i)}{\rho^*} \tag{6}$$

As shown in Eq. 6, ρ^* represents the highest centrality score in the semantic block, and $\rho(b_i)$ represents the centrality score of semantic block b_i. After calculating the retention ratio η for each block, we use BERTSum [15] to score each sentence within the block and retain the top η sentences. Blocks with higher centrality scores retain more sentences. Finally, the selected sentences x_i are concatenated, and a fine-tuned BART [9] model generates the text summary $S = \{y_1, \ldots, y_t, \ldots, y_l\}$.

3.5 Image Selection

In order to more accurately select the image that is closely related to the generated text summary, we first use the pre-trained BERT [4] model to encode the image captions and the generated text summary. The result of this projection is that we obtain the feature vector representations of the text summary and the image features, which align them in the same textual semantic space. Next, we use the cosine similarity to calculate the semantic similarity score for each pair of caption-text vectors, which characterizes their relative position and similarity in the semantic space. This process enables us to accurately judge the degree of similarity between each image and the generated text summary, thereby selecting the most relevant image for the multimodal summarization task (Fig. 3).



Nick Buoniconti, 76, was a legend with the Miami Dolphins. He was diagnosed with dementia in May, and his doctors now believe he also suffers CTE. On Friday, Buoniconti announced he is leaving his brain with Boston University. The famed brain research team diagnosed CT posthumously in the late Patriots star Aaron Hernandez. Last month, Buoniconti's son Marc, who was left paralyzed for life after a clash in a college football game 32 years ago, said youth football should be banned.

Fig. 3. Image Selection Strategy

4 Experimental Settings

4.1 Dataset

During the experimental process, the data was referenced to the dataset MSMO [27], which included 293,965 training data, 10,355 validation pairs, and 10,261 test pairs. The MSMO dataset is constructed using a corpus from the Daily Mail website paired with multiple images and utilizes the manually written highlights provided by the Daily Mail as reference text summaries. To obtain the pictorial references for the test set, MSMO employs 10 graduate students to select relevant images from the articles corresponding to each reference text summary.

4.2 Evaluation Metrics

We choose the following evaluation metrics:

ROUGE-{1,2,L} is used as the standard evaluation metric for automatic summarization. It measures the similarity between two text passages, typically focusing on the overlap of n-grams, word sequences, and word pairs.

Image Precision (IP) is a commonly used metric for evaluating image selection performance. It defines image accuracy by calculating the ratio between the properly recommended images and the reference ones. Used to measure the ability to accurately select images with high correlation when given a reference image.

M_{sim}, as a metric for evaluating the correlation between images and texts, is achieved by calculating the maximum similarity between each sentence in the final summary and the image.

4.3 Baselines

To demonstrate the effectiveness of our proposed method, we compared it with various summarization methods.

BERTSum [15] is a unimodel for extractive and abstractive text summarization based on BERT, with two variants of BertAbs (abstractive) and BertExtAbs (hybrid).

BART [9] is a pre-trained model composed of a bidirectional encoder and an autoregressive decoder that can better understand the complex context relationship.

ATG/ATL/HAN [27] uses the global, local and hierarchical image features, respectively, for the multimodal abstractive summarization task.

MOF [28] incorporate a multimodal objective function into ATG. Out of the four variants of the multimodal objective function, we select the two that exhibit better performance we utilized.

UniMS [24] propose a unified framework for multimodal summarization, which introduces a visual guided decoder to better integrate textual and visual modalities.

ViL-Sum [3] model paragraph-level vision-language semantic alignment for better learning multi-modal semantics.

5 Experimental Results

5.1 Automatic Evaluation

We split the performance experiments into automatic evaluation and human evaluation to better analyze the detailed impact of our method. Table 1 summarizes the automatic evaluation results on abstractive summarization and image selection subtasks. The first block in the table includes abstractive summarization methods with text-only input, while the second block includes abstractive methods with multimodal input. By investigating the results, we make the following observations:

Table 1. Main results of different metrics. R-1, 2, L refers to ROUGE-1, 2, L, IP refers to Image Precision. Noted that the experimental results of other methods are taken from UniMS [24].

Model	R-1	R-2	R-L	IP	$\mathrm{M}_{\mathrm{sim}}$
Text Abstractive					
BertAbs [15]	39.02	18.17	33.20	–	–
BertExtAbs [15]	39.88	18.77	38.36	–	–
BART [9]	41.83	19.83	39.74	–	–
Multimodal Abstractive					
ATG [27]	40.63	18.12	37.53	59.28	25.82
ATL [27]	40.86	18.27	37.75	62.44	13.26
HAN [27]	40.82	18.30	37.70	61.83	12.22
$\mathrm{MOF}^{\mathrm{RR}}_{\mathrm{enc}}$ [28]	41.05	18.29	37.74	62.63	26.23
$\mathrm{MOF}^{\mathrm{RR}}_{\mathrm{dec}}$ [28]	41.20	18.33	37.80	65.45	26.38
UniMS [24]	42.94	20.50	40.96	69.38	29.72
ViL-Sum [3]	44.29	**20.96**	41.34	66.27	32.17
Our Method	**44.80**	20.32	**41.46**	**74.19**	**32.35**

We achieve the best IP and M_{sim} compared with other methods due to the implementation of an Image Selection strategy (i.e., our method outperforms UniMS by achieving 6.9% and 8.8% higher scores on the IP and M_{sim} metrics, respectively). Compared with existing methods that directly select the image in the model, our method projects both image captions and text summary generated from the document into the same textual semantic space for comparison and then selects the most relevant image caption compared with the generated text summary, where the corresponding image is selected as the image summary. The results illustrate that the image selection strategy enhances the coherence between the textual and visual information to achieve higher performance than existing methods.

In the first block of text-only models, we find that the fine-tuned pre-trained BART model exhibits up to 9.7% higher ROUGE-L score than $\text{MOF}^{\text{RR}}_{\text{enc}}$, which indicates the powerful summarization capabilities of BART. Compared with UniMS, BART also achieves competitive results, which only decrease by 1.11, 0.67, and 1.22 under ROUGE-{1,2,L} scores, respectively.

In the second block, previous multimodal methods (such as ATL [27] and MOF [28]) demonstrate no capability over the single-modal text summarization model as mentioned. These results concluded that too many images could bring noise, and the long document contains enough information for text generation. By using a more powerful unified model, UniMS improves the ROUGE-L score and IP score by 3.16 and 3.39 over $\text{MOF}^{\text{RR}}_{\text{enc}}$, respectively. ViL-Sum, on the other size, models paragraph-level vision-language alignment for summarization. In contrast, our method utilizes a sentence-level filtering strategy capable of filtering irrelevant information within the facet-aware semantic blocks. By employing our designed modules, we capture useful information and generate more accurate summaries. As a result, our method achieves higher ROUGE scores compared to other methods. Meanwhile, the proposed strategy for text-caption alignment can integrate visual information into the same textual semantic space and thereby accurately select the image, achieving up to 11.9% over ViL-Sum under IP score.

M_{sim} is used to check the text-image relevance of the pictorial summary. Our method projects both image captions and the generated text summary into the same textual semantic space for comparison. The results show that we achieve 32.35 under M_{sim}, which is higher than 29.72 obtained by UniMS. Thus, we conclude that the text-image relevance is much higher than the state-of-the-art UniMS model as we expected, which demonstrates the effectiveness of the image selection strategy.

5.2 Human Evaluation

We present the human evaluation results of the method we proposed. For this purpose, we engaged three part-time graduate students from the Alibaba Crowdsourcing platform to assess the multimodal summaries generated by our best-performing model. We selected the most representative models, ATG and UniMS, for comparison with our approach. We randomly selected 200 samples from the test dataset for evaluation and instructed them to judge the multimodal summary based on the following criteria:

- Coverage (Cov): Compare the model-generated text summary with the actual text summary to check if the main points are adequately covered.
- Grammar (Gra): Examine whether the model-generated text summary is grammatically correct.
- Consistency (Con): Measure the factual alignment between the summary and the source document.
- Image-Relevance (IR): Indicate the text-image relevance of multimodal outputs.

The evaluators rated the selected samples on a scale from 1 (the worst) to 5 (the best) for each of the four dimensions. The average results are presented in Table 2. Our approach outperforms the comparative systems in all metrics, with the lowest score exceeding 0.34. The voters generally agreed that our system's summary exhibit higher Cov and Con scores, reflecting factual consistency between the summary and the original article. Furthermore, we achieved the highest IR score, indicating a strong correlation between the images and the generated text summary. Additionally, the text summary is predominantly grammatically and semantically correct. These results provide further validation of the effectiveness of our proposed method.

Table 2. Human evaluation results.

Model	Cov	Gra	Con	IR
ATG [27]	3.64	4.14	3.87	3.72
UniMS [24]	3.91	4.25	4.09	4.14
Our Method	**4.23**	**4.36**	**4.28**	**4.32**

5.3 Ablation Experiment

We set up an ablation study to validate the effectiveness of our method, investigating the impact of three key modules, including Text-Caption Alignment (TCA), Facet-Aware Document Segmentation (FADS) and Salient Information Retrieval (SIR). Our experimental setup is described as follows, with the results shown in Table 3.

Table 3. Ablation study.

Approaches	R-1	R-2	R-L	IP	$\mathrm{M_{sim}}$
Our method	**44.80**	**20.32**	**41.46**	**74.19**	32.35
w/o TCA	43.76	19.40	40.43	74.01	32.27
w/o FADS	44.55	20.15	41.20	74.12	32.34
w/o SIR	44.73	20.28	41.36	74.15	**32.40**

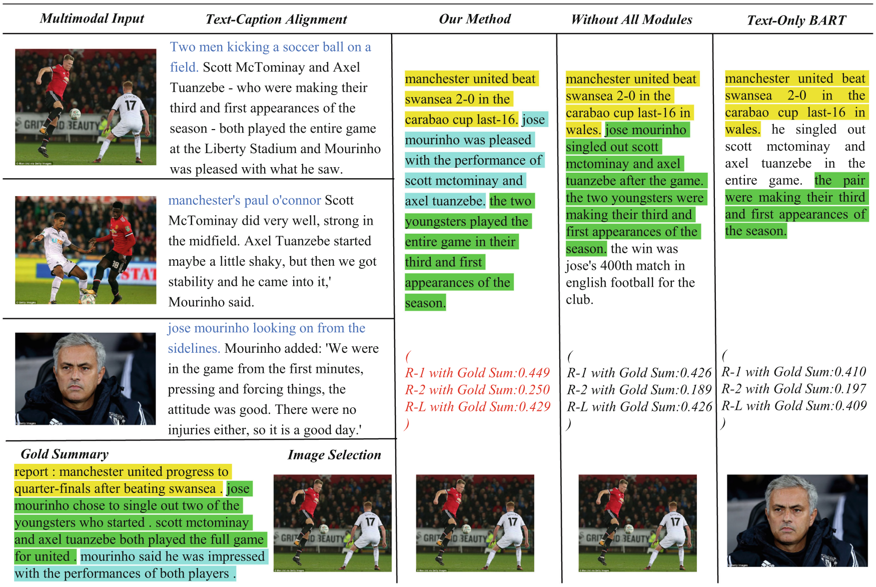
Impact Analysis of TCA. Our TCA strategy significantly outperforms existing methods, as shown in Table 3. TCA is crucial for enhancing information amalgamation and cohesiveness between textual and visual elements. Without TCA, the lower ROUGE scores and reduced M_{sim} metric indicate inferior congruence between the generated summary and the reference text, as well as weakened semantic alignment between textual and visual components. Inappropriately appending image captions to the text may lead to redundancy or incoherence in the generated summary. Therefore, our TCA strategy is essential for selecting the optimal position for visual information insertion.

Impact Analysis of FADS. Under the condition without the FADS module, we select n sentences as a block, where n represents the average number of sentences per paragraph. The results show that incorporating FADS significantly improves summarization performance, with notable advantages in ROUGE scores, IP metric, and M_{sim} metric. This highlights the crucial role of the FADS module in shaping the summarization process. The substantial differences in ROUGE scores indicate that FADS has a significant impact on the informativeness and coherence of the generated summary. The decrease in the IP value suggests that the SIR module, under the influence of FADS, contributes to selecting more relevant sentences for summarization. Moreover, the slight improvement in the M_{sim} metric implies that FADS helps maintain the alignment between visual and textual elements, even when the optimal text summary is not selected.

Impact Analysis of SIR. Removing the SIR module means that no sentence filtering is performed on the blocks, and the text obtained after the TCA strategy is directly fed into the summarization model. As shown in Table 3, this leads to lower ROUGE scores, indicating that the noise in the input document negatively affects the summarization quality. However, the minor changes in IP and M_{sim} suggest that our image selection remains accurate, demonstrating the effectiveness of our method in matching visual and textual information. The SIR strategy plays a crucial role in our approach by reducing noise and enhancing information fusion, ultimately improving multimodal text summarization.

5.4 Case Study

Figure 4 presents a randomly selected sample consisting of multiple paragraphs of text and images. The blue-marked part of the figure represents the results obtained after processing the sample with the alignment module. These results clearly demonstrate our strategy is capable of completing the cross-modal data fusion by locating the image captions in the suitable insertion position. For example, adding "*Jose Mourinho looking on from the sidelines.*" before "*Mourinho added ...*" provides extra context. Observing the results, comparing with the other approaches, our method incorporates all the key aspects, which are shown in the left corner as the *Gold Summary* marked with three different colors. Specifically, our method can generate the highlighted blue sections, while comparison systems do not. Our method and the without-all-modules method select the same image as the gold summary and achieve better ROUGE scores than Text-Only BART. Moreover, as shown in the Fig. 4, text-only input led to incorrect image selection, which demostrates that incorporating image captions



Multimodal Input | *Text-Caption Alignment* | *Our Method* | *Without All Modules* | *Text-Only BART*

Two men kicking a soccer ball on a field. Scott McTominay and Axel Tuanzebe - who were making their third and first appearances of the season - both played the entire game at the Liberty Stadium and Mourinho was pleased with what he saw.

manchester's paul o'connor Scott McTominay did very well, strong in the midfield. Axel Tuanzebe started maybe a little shaky, but then we got stability and he came into it,' Mourinho said.

jose mourinho looking on from the sidelines. Mourinho added: 'We were in the game from the first minutes, pressing and forcing things, the attitude was good. There were no injuries either, so it is a good day.'

Our Method: manchester united beat swansea 2-0 in the carabao cup last-16. jose mourinho was pleased with the performance of scott mctominay and axel tuanzebe. the two youngsters played the entire game in their third and first appearances of the season.

(R-1 with Gold Sum:0.449 R-2 with Gold Sum:0.250 R-L with Gold Sum:0.429)

Without All Modules: manchester united beat swansea 2-0 in the carabao cup last-16 in wales. jose mourinho singled out scott mctominay and axel tuanzebe after the game. the two youngsters were making their third and first appearances of the season. the win was jose's 400th match in english football for the club.

(R-1 with Gold Sum:0.426 R-2 with Gold Sum:0.189 R-L with Gold Sum:0.426)

Text-Only BART: manchester united beat swansea 2-0 in the carabao cup last-16 in wales. he singled out scott mctominay and axel tuanzebe in the entire game. the pair were making their third and first appearances of the season.

(R-1 with Gold Sum:0.410 R-2 with Gold Sum:0.197 R-L with Gold Sum:0.409)

Gold Summary

report : manchester united progress to quarter-finals after beating swansea . jose mourinho chose to single out two of the youngsters who started . scott mctominay and axel tuanzebe both played the full game for united . mourinho said he was impressed with the performances of both players .

Image Selection

Fig. 4. A Case Study of our work. The blue text in the Text Caption Alignment section refers to the corresponding image captions generated. The highlighted text in yellow, green, and blue is the content part covered in the Gold Summary, and the corresponding color on the right is the comparison between the three methods and the Gold Summary. (Color figure online)

is crucial for accurate image selection by addressing information imbalance in multimodal summarization.

6 Conclusion

In this paper, we propose a novel multimodal summarization method that utilizes image captions to fusion the semantic information between visual and textual modalities. Our method segments the document into various facet-aware blocks and meticulously filters out irrelevant content within each block to reduce the loss of key information and generate a more precise summary. Additionally, we also propose a straightforward yet effective image selection strategy, which largely bridges the gap between the pictorial summary. The experimental results demonstrate that the proposed model outperforms existing methods on both automatic metrics and manual evaluation. However, our method suffers from handling the long document, which is also a challenge in existing multimodal summarization. In future work, we will focus on improving the performance of processing a long document in the context of multimodal summarization. Moreover, we will also devote ourselves to refining these techniques, compressing the model, or creating a broader applicable dataset.

Acknowledgements. This work was supported in part by the Beijing Municipal Science & Technology Commission under Grant Z231100001723002, and in part by the National Key Research and Development Program of China under Grant 2020YFB1406702-3, in part by the National Natural Science Foundation of China under Grant 61772575 and 62006257.

References

1. Cao, B., Araujo, A., Sim, J.: Unifying deep local and global features for image search. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020, Proceedings, Part XX 16, pp. 726–743. Springer (2020)
2. Chen, J., Zhuge, H.: Extractive text-image summarization using multi-modal RNN. In: 14th International Conference on Semantics, Knowledge and Grids, SKG 2018, Guangzhou, China, 12-14 September 2018, pp. 245–248. IEEE (2018). https://doi.org/10.1109/SKG.2018.00033
3. Cui, C., Liang, X., Wu, S., Li, Z.: Align vision-language semantics by multi-task learning for multi-modal summarization. Neural Comput. Appl. 1–14 (2024)
4. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2-7 June 2019, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/n19-1423
5. Ding, C., et al.: Learning aligned audiovisual representations for multimodal sentiment analysis. In: Ghosh, S., Dhall, A., Kollias, D., Goecke, R., Gedeon, T. (eds.) Proceedings of the 1st International Workshop on Multimodal and Responsible Affective Computing, MRAC 2023, Ottawa, ON, Canada, 29 October 2023, pp. 21–28. ACM (2023). https://doi.org/10.1145/3607865.3613184
6. Evangelopoulos, G., et al.: Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention. IEEE Trans. Multim. **15**(7), 1553–1568 (2013). https://doi.org/10.1109/TMM.2013.2267205
7. Hearst, M.A.: Text tiling: segmenting text into multi-paragraph subtopic passages. Comput. Linguist. **23**(1), 33–64 (1997)
8. La, T., Tran, Q., Tran, T., Tran, A., Dang-Nguyen, D., Dao, M.: Multimodal cheapfakes detection by utilizing image captioning for global context. In: Dao, M., Dang-Nguyen, D., Riegler, M. (eds.) ICDAR@ICMR 2022: Proceedings of the 3rd ACM Workshop on Intelligent Cross-Data Analysis and Retrieval, Newark, NJ, USA, 27–30 June 2022, pp. 9–16. ACM (2022). https://doi.org/10.1145/3512731.3534210
9. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020, pp. 7871–7880. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.acl-main.703

10. Li, H., Zhu, J., Ma, C., Zhang, J., Zong, C.: Multi-modal summarization for asynchronous collection of text, image, audio and video. In: Palmer, M., Hwa, R., Riedel, S. (eds.) Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 September 2017, pp. 1092–1102. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/d17-1114
11. Li, H., Zhu, J., Ma, C., Zhang, J., Zong, C.: Read, watch, listen, and summarize: multi-modal summarization for asynchronous text, image, audio and video. IEEE Trans. Knowl. Data Eng. **31**(5), 996–1009 (2019). https://doi.org/10.1109/TKDE.2018.2848260
12. Li, J., Li, D., Xiong, C., Hoi, S.C.H.: BLIP: bootstrapping language-image pre-training for unified vision-language understanding and generation. In: Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., Sabato, S. (eds.) International Conference on Machine Learning, ICML 2022, 17–23 July 2022, Baltimore, Maryland, USA. Proceedings of Machine Learning Research, vol. 162, pp. 12888–12900. PMLR (2022)
13. Liang, X., Li, J., Wu, S., Zeng, J., Jiang, Y., Li, M., Li, Z.: An efficient coarse-to-fine facet-aware unsupervised summarization framework based on semantic blocks. In: Calzolari, N. (eds.) Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, 12–17 October 2022, pp. 6415–6425. International Committee on Computational Linguistics (2022)
14. Liang, Y., Meng, F., Wang, J., Xu, J., Chen, Y., Zhou, J.: D^{2}tv: dual knowledge distillation and target-oriented vision modeling for many-to-many multimodal summarization. In: Bouamor, H., Pino, J., Bali, K. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023, pp. 14910–14922. Association for Computational Linguistics (2023)
15. Liu, Y., Lapata, M.: Text summarization with pretrained encoders. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019, pp. 3728–3738. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1387
16. Pal, A., Wang, J., Wu, Y., Kant, K., Liu, Z., Sato, K.: Social media driven big data analysis for disaster situation awareness: a tutorial. IEEE Trans. Big Data **9**(1), 1–21 (2023). https://doi.org/10.1109/TBDATA.2022.3158431
17. Palaskar, S., Libovický, J., Gella, S., Metze, F.: Multimodal abstractive summarization for how2 videos. In: Korhonen, A., Traum, D.R., Màrquez, L. (eds.) Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers, pp. 6587–6596. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/p19-1659
18. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18–24 July 2021, Virtual Event. Proceedings of Machine Learning Research, vol. 139, pp. 8748–8763. PMLR (2021)
19. Ruan, Q., Ostendorff, M., Rehm, G.: Histruct+: Improving extractive text summarization with hierarchical structure information. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, 22–27 May 2022, pp. 1292–1308. Association for

Computational Linguistics (2022). https://doi.org/10.18653/V1/2022.FINDINGS-ACL.102

20. Thuma, B.S., de Vargas, P.S., Garcia, C.M., de Souza Britto Jr., A., Barddal, J.P.: Benchmarking feature extraction techniques for textual data stream classification. In: International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, 18–23 June 2023, pp. 1–8. IEEE (2023). https://doi.org/10.1109/IJCNN54540.2023.10191369
21. Wu, Q., Shen, C., Wang, P., Dick, A.R., van den Hengel, A.: Image captioning and visual question answering based on attributes and external knowledge. IEEE Trans. Pattern Anal. Mach. Intell. **40**(6), 1367–1381 (2018). https://doi.org/10.1109/TPAMI.2017.2708709
22. Yang, H., Zhao, Y., Qin, B.: Face-sensitive image-to-emotional-text cross-modal translation for multimodal aspect-based sentiment analysis. In: Goldberg, Y., Kozareva, Z., Zhang, Y. (eds.) Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, 7–11 December 2022, pp. 3324–3335. Association for Computational Linguistics (2022). https://doi.org/10.18653/V1/2022.EMNLP-MAIN.219
23. Zhang, C., Zhang, Z., Li, J., Liu, Q., Zhu, H.: Ctnr: compress-then-reconstruct approach for multimodal abstractive summarization. In: International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, 18–22 July 2021, pp. 1–8. IEEE (2021). https://doi.org/10.1109/IJCNN52387.2021.9534082
24. Zhang, Z., Meng, X., Wang, Y., Jiang, X., Liu, Q., Yang, Z.: Unims: a unified framework for multimodal summarization with knowledge distillation. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, 1 March 2022, pp. 11757–11764. AAAI Press (2022)
25. Zhang, Z., Shu, C., Chen, Y., Xiao, J., Zhang, Q., Lu, Z.: ICAF: iterative contrastive alignment framework for multimodal abstractive summarization. In: International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, 18–23 July 2022, pp. 1–8. IEEE (2022). https://doi.org/10.1109/IJCNN55064.2022.9892884
26. Zheng, H., Lapata, M.: Sentence centrality revisited for unsupervised summarization. In: Korhonen, A., Traum, D.R., Màrquez, L. (eds.) Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers, pp. 6236–6247. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/p19-1628
27. Zhu, J., Li, H., Liu, T., Zhou, Y., Zhang, J., Zong, C.: MSMO: multimodal summarization with multimodal output. In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018, pp. 4154–4164. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/d18-1448
28. Zhu, J., Zhou, Y., Zhang, J., Li, H., Zong, C., Li, C.: Multimodal summarization with guidance of multimodal reference. In: The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, 7–12 February 2020, pp. 9749–9756. AAAI Press (2020)

Word-Diffusion: Diffusion-Based Handwritten Text Word Image Generation

Aniket Gurav[1](✉), Narayanan C. Krishnan[2], and Sukalpa Chanda[1,2]

[1] Department of Computer Science and Communication,
Østfold University College, 1757 Halden, Norway
aniketag@hiof.no, sukalpa@ieee.org

[2] Department of Data Science, Indian Institute of Technology Palakkad, Kerala, India
ckn@iitpkd.ac.in

Abstract. Generating realistic handwritten word images that closely resemble a target style remains a challenging task in document image analysis. In recent years, deep learning techniques, such as Latent Diffusion Models (LDM), have shown promise in generating styled handwritten text. However, these models face significant challenges when creating images for 'Out of Vocabulary' (OOV) words, impacting their overall effectiveness. In this paper, we introduce an extended diffusion-based Handwritten generation method that incorporates a novel conditioning mechanism. It is based on the Pyramidal Histogram of Shapes (PHOS) representation, which takes into account the spatial and structural characteristics of the target handwriting style. By conditioning the diffusion model on input text, PHOS vector, and writer ID, our approach enables the generation of handwritten word images. Notably, our approach outperforms the original diffusion model, which only uses text and writer ID as conditions, in generating both in-sample and out-of-sample. Furthermore, we have developed a faster inference method that significantly reduces the number of steps required for generating the output. Through qualitative and quantitative evaluations, we demonstrate the effectiveness of our proposed method.

Keywords: Denoising diffusion probabilistic model · Handwritten Text Recognition · Synthetic Handwritten Data

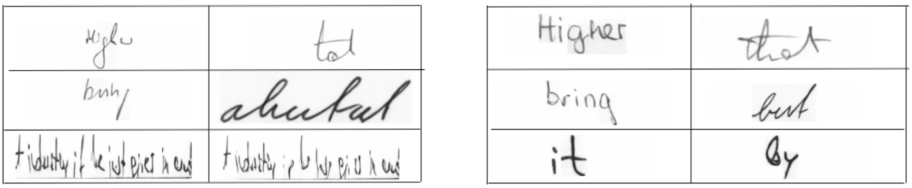
1 Introduction

Handwritten text generation has emerged as a prominent research area within the field of document image analysis. Its applications are wide-ranging, encompassing data augmentation for handwriting recognition systems and the creation of personalized digital content. Deep learning (DL) techniques have achieved remarkable performance in many domains, including handwriting recognition. However, to fully realize their potential, these techniques demand extremely

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 53–72, 2025.
https://doi.org/10.1007/978-3-031-78495-8_4

large volumes of high-quality annotated data. DL methods rely heavily on learning from large quantities of data to effectively discern patterns and generalize capabilities, thereby enhancing their generalization and performance. Acquiring a large annotated dataset, however, can be challenging and resource-intensive, presenting a significant obstacle in practice. Without access to sufficient annotated data for training, these approaches often struggle to attain top accuracy levels. Data generation techniques offer a solution to this critical need. They enable the automatic synthesis of high-quality additional training examples at scale, supplementing limited real data. Our research focuses on generating handwritten text data, with a specific emphasis on two key aspects: maintaining the style of known writers and generating both known and unknown words that the synthetic data generation system has never encountered. This sets our approach apart by enabling us to generate text that not only preserves the desired style of a known writer but also extends the system's capability to generate novel words it has never encountered. By pursuing this direction, we aim to provide a comprehensive solution that combines style preservation with the generation of contextually diverse and stylistically consistent text. This ultimately enhances the practicality and versatility of data generation techniques in the field of handwritten text generation.

Recent advancements in deep learning, particularly with the introduction of Generative Adversarial Networks (GANs) [2,9] and LDMs like WordStylist [18], have paved the way for generating realistic and stylized handwritten text images. Current state-of-the-art methods for handwritten text generation, such as GANwriter [13], SmartPatch [17] and WordStylist [18], has demonstrated impressive results in generating visually appealing handwritten words conditioned on both text content and style information. However, the performance current state-of-the-art methods in generating OOV word images have not been quantitatively evaluated. OOV word images refer to the ability to generate handwritten words that were not included in the training dataset. While current state-of-the-art methods excel at replicating known words and maintaining the style, their performance in generating novel, previously unseen words is not promising. This limitation hinders the practical applicability of these methods in scenarios where only a few word classes are represented in the dataset. GANwriter [13] employs a conditional GAN architecture to generate handwritten word images. In contrast, WordStylist [18] leverages the power of LDMs to generate styled handwritten text by conditioning the model on both text content and style vectors. While both approaches have shown promising results, WordStylist [18] has demonstrated superior performance in terms of accuracy while maintaining the writing style. The exceptional performance of LDM has been observed in other domains [6]. To improve in performance metrics, it is essential to generate high-quality synthetic data using a framework and integrate it into the recognition system alongside the original data. However, it is crucial to ensure that the generated data is contextually accurate to avoid any degradation in the performance of the recognition system. When incorporating synthetic data into the recognition system, it is crucial to ensure that the generated samples closely resemble the characteristics and variations present in the real data. This includes accurately capturing the textual content, handwriting style, and other relevant features. If the synthetic data deviates significantly from the contextually accurate representation, it may negatively impact the performance of the recognition system. We observed several such cases with WordStylist [18]. Figure 1a illustrates training data samples generated using the officially provided WordStylist weights. In the first row, the framework attempts to generate the words "Higher" and "that" exhibiting a partial resemblance to the expected text. Similarly, in the second row of the first case, the generated word "bring" also bears partial similarity to the expected text. However, these generated words are difficult for humans to read due to their lack of clarity. Notably, the framework fails to generate the second word in the second row, where the expected text is "but" The generated output does not resemble the intended text at all, resulting in a complete failure of the framework in this case. The third row of the diagram showcases the generation of unreadable words by the framework. The expected text in this row includes the words "it" and "by" However, the generated words do not exhibit any clear resemblance to the expected text, making them illegible. These failure cases highlight the limitations and challenges faced by the WordStylist [18] framework in accurately generating desired word images, particularly in terms of readability and reliable generation of the expected text. Similar types of samples generated by the diffusion model may hamper the performance metric when used alongside the original training data.



(a) WordStylist generated images

(b) DiffWord generated images

Fig. 1. Images generated by Wordstylist and DiffWord framework

To address this limitation, we propose an extension to the WordStylist [18] model, which we refer to as "DiffWord". We incorporate a novel conditioning mechanism based on the PHOS representation [1] along with text and writer ID. The PHOS is a compact and discriminative representation that encodes the spatial distribution of characters and shapes within the handwritten text image. By conditioning the diffusion model on both the text content and the PHOS representation, we aim to capture the spatial and structural characteristics of the target handwriting style more effectively. In Fig. 1b, we present the same word examples as shown in Fig. 1a, generated using the DiffWord method. The images generated by DiffWord in Fig. 1b and 2 exhibit improved clarity, readability, and accuracy compared to those generated by the WordStylist [18] method (Fig. 1a). These differences are evident in the finer details, enhanced legibility, and overall visual quality of the DiffWord-generated word examples, making them clearer and more visually appealing. Our proposed approach leverages the U-Net architecture [21] as the backbone of the diffusion model, similar to WordStylist [18].

The performance of the recognition [20] system also depends on the amount of quality data provided. Therefore, the frameworks must generate high-quality synthetic data in a timely manner. The method “WordStylist” [18] is based on LDM approach that takes 600 to 1000 steps during the generative sampling procedure, which is computationally expensive and time-consuming. To address the computational cost and time required, we introduce an early-sampling technique that significantly reduces the number of steps needed. Our proposed method generates high-quality synthetic handwritten text data in just 120 steps for IAM [16] dataset and in 200 steps for the CVL dataset [14], which is a significant improvement over the 600 to 1000 steps required by the “WordStylist” [18] method. This technique enables us to generate a large amount of data with fewer steps, thus greatly enhancing the efficiency of the generation process. We have also developed a validation framework that utilizes a recognition model to verify the accuracy of the generated images in terms of their textual content. If the generated images pass the validation, they are considered correct and added to the synthetic data.

The main contributions of our work are as follows:

1. We extend the WordStylist [18] model by incorporating the PHOS conditioning branch, enabling the generation of handwritten word images that better capture the spatial and structural characteristics of the target style.
2. We propose an early sampling technique that significantly reduces the number of steps required during the generative diffusion sampling procedure from 600 to just 120 steps for IAM [16] dataset and 1000 to 200 for CVL [14] dataset.

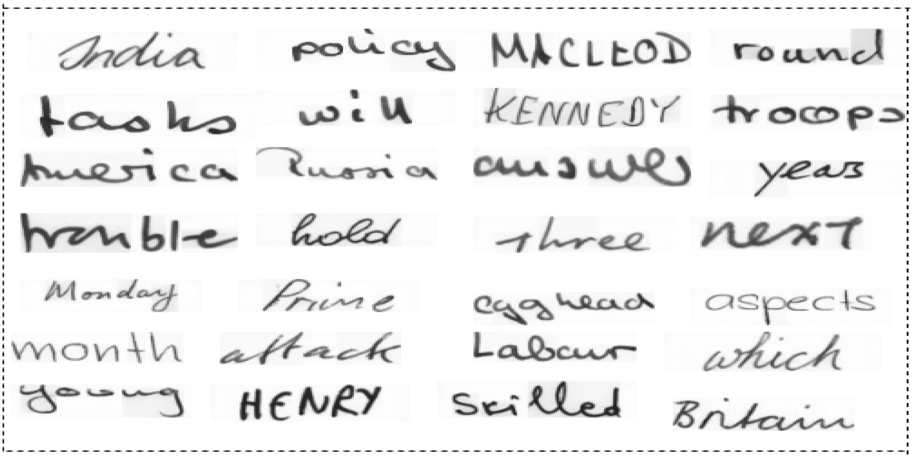


Fig. 2. Words generated by DiffWord framework

3. We conduct a quantitative evaluation of OOV word generation, providing insights through detailed metrics, This analysis establishes a baseline for OOV word performance, highlighting our model's ability to effectively generate and assess handwriting styles for words not present in the training dataset.

The structure of this paper is as follows: Sect. 2 reviews pertinent literature in handwritten text generation. Section 3 provides an overview. Our proposed methodology is detailed in Sect. 4. Section 5 outlines the experimental framework, datasets, and evaluation criteria, and offers a discussion of the findings. Limitations and challenges are discussed in Sect. 5.6. The paper is concluded in Sect. 6.

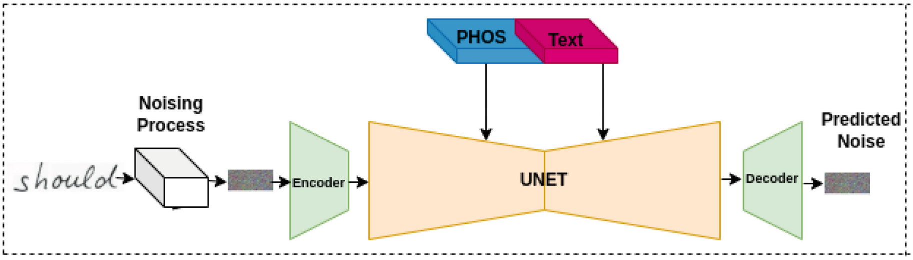


Fig. 3. DiffWord Architecture

2 Related Work

In recent years, several techniques have been proposed for synthetic text generation, specifically in the domain of word-level generation. In this section, we discuss the relevant works related to handwritten word generation. Graves [10] investigated synthesizing online handwriting trajectories of English texts using RNN. The authors of [27] extended the [11] approach for online handwriting synthesis. Several recent works have explored the use of GANs [24,28] for synthetic handwritten text generation. GAN-based approaches, such as GANwriting [13] and HiGAN+ [9], have shown promising results in generating diverse and authentic word images. GANwriting [13] leverages calligraphic style features and textual content to intricately condition the generative process. The HiGAN+ model introduced in [9] improved upon prior work by conditioning generative image synthesis on both disentangled representations of calligraphic style and textual content, employing contextual and local patch losses to enhance style consistency and image quality, and leveraging a more compact architecture based on reusing early writer identification layers. In TS-GAN [4], the ability to extract styles from images via pixel-level reconstruction was demonstrated. While ScrabbleGAN presented in [7] had the ability to synthesize handwritten text of arbitrary length by concatenating individual letter images, it was limited in its ability to accurately mimic the calligraphic styles. In [26], JokerGAN was introduced, a memory-efficient model for generating handwritten text with awareness of the text line structure. These methods utilize adversarial training frameworks and conditioning on input text and style features to produce coherent word sequences. However, challenges persist in achieving perfect stylistic coherence and inter-character spacing. Diffusion models like WordStylist [18] have also demonstrated potential for word generation conditioned on text, but do not explicitly model global stylistic cues. Transformer-based models, such as Handwriting Transformers (HWT) [2] and VATr [19], have also been proposed for stylized word generation. These models employ transformer encoders and decoders to capture global and local handwriting styles and enable style-content entanglement at the character level. However, they do not integrate representations capturing the intrinsic hierarchical structure of handwriting. HiGAN [8] introduced an improved GAN architecture for disentangled handwriting style generation conditioned on text. It achieved state-of-the-art performance but relied on an unstable GAN training framework and limited style representations, constraining generalization. SLOGAN [15] proposed a method for synthesizing parameterized and controllable handwriting styles for arbitrary-length text, including out-of-vocabulary words. While it demonstrated strong stylistic control, ensuring perfect coherence when varying generation parameters posed difficulties. These works made progress in synthesizing longer handwritten sequences, with ScrabbleGAN [7] achieving sentences and HiGAN/SLOGAN extending to arbitrary lengths. However, challenges remained in maintaining perfect spatial characteristics and coherence over long sequences, motivating our exploration of leveraging holistic PHOS representations to better capture these global style cues during word generation. The GlyphControl [25] technique utilizes diffusion models to digitally render text guided by glyph maps. Conditioning on glyph templates during inference enables consistent custom text synthesis. However, the generated text is not handwritten. In our work, we propose leveraging the PHOS [1] technique, which extracts multi-scale histograms encoding the statistical distribution of shape and character features. By analyzing this hierarchical representation, our model aims to capture fine-grained stylistic details and variations in handwriting for improved word generation. To the best of our knowledge, our work is the first to explore word-level generation using the PHOS technique in the context of synthetic text generation.

3 Overview

Denoising Diffusion Probabilistic Models (DDPMs) aim to reconstruct the original data inputs. They accomplish this by learning the inverse operation of gradually adding noise onto data instances across multiple time periods. The methodology is founded upon concepts from the field of thermodynamics, which studies the transfer of heat and energy in systems [12,22]. DDPMs utilize a process of incremental noising via a Markov chain. The models are trained to minimize the effects of noise contamination by optimizing the reverse mechanism. Once completed, this enables the generation of novel data representations through chained synthesis from noise components and recovering the underlying uniformities across many iterations of noise subtraction. The goal of this modeling approach is to reconstruct source data inputs from corrupted latent representations produced during a forward diffusion phase. In the forward process, the original data undergoes a stepwise transformation. At sequential intervals numbered 1 through T, random perturbations in the form of Gaussian noise are systematically added to and combined with the data. This incorporation of noise leads to the derivation of a chain of latent representations $x_1, x_2, ..., x_T$. The magnitude of noise included at each interval is governed by a designated noise parameter scale, β_t, where β_t can scale between 0 and 1. These derived latents are interconnected by transitional rules that bridge adjacent time lapses, facilitating a progressive evolution from the original data x_0 into the ultimate latent x_T through the sequential introduction of varying noise as shown in Eq. 1.

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{1}$$

DDPMs learn to recover the original information x_0 by removing the extra noise added earlier. A neural network is trained for this task. It starts with the hidden version that has changed the most over time. The network repeats undoing changes step-by-step from t = T to t = 1. First, it guesses what the version looked like before more noise was added, $p_\theta(x_{t-1}|x_t)$ as shown in Eq. 2. Then it uses this guess to help undo the next step. This process takes it back one small change at a time. It keeps fixing previous versions x_{t-1} based on later ones x_t. After many repeats of $p_\theta(x_{t-1}|x_t)$, the network learns by making its guesses p_θ closer to the real previous versions.

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \quad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \mathbf{I})) \tag{2}$$

$$L = E_{x_0,t,\epsilon}[||\epsilon - \epsilon_\theta(x_t, t)||^2] \tag{3}$$

During the training process, the loss function, defined in Eq. 3, quantifies the mean squared error between the actual noise ϵ introduced at time step t and the noise estimated by the model ϵ_θ.

4 Methodology

This section describes our proposed methodology for generative diffusion models of handwritten text images. The training procedure of a conditional generative diffusion model is discussed first. We then discuss two approaches for sampling from the trained model: (a) a conventional method following the complete reverse process, and (b) an efficient early sampling technique. In the conventional method, an exact copy of the data used for training a diffusion model is generated, preserving the same writer ID and text associated with each image ID. However, with early sampling, a different strategy is employed. The same training split is generated, but the writer IDs associated with each image ID change while keeping the text unchanged. Finally, we present a validation framework for assessing the quality of images generated with early sampling by performing text recognition from the images.

4.1 Training

Figure 3 illustrates the architecture of DiffWord. Our proposed method builds upon the LDM architecture introduced in the WordStylist paper [18]. The input image initially passes through a pre-trained Variational Encoder (VE), generating a low-dimensional latent representation of the image. This latent encoding reduces the image's dimensionality while retaining important semantic information. The forward diffusion process gradually corrupts the image's latent representation by adding Gaussian noise at each timestep. A noise scheduler is employed to gradually increase the level of noise. The noise level is linearly incremented from an initial value of $\beta_1 = 10^{-4}$ to a final value of $\beta_{600} = 0.02$ over a total of $T = 600$ timesteps for IAM [16] data and $T = 1000$ for CVL [14] dataset. This incremental increase ensures that the diffusion process effectively spreads and incorporates noise throughout the image's latent representation. Our diffusion model employs a U-Net [21] architecture as its backbone. The U-Net receives several inputs, including the noisy image latents, the corresponding timestep, and the desired conditions. These conditions consist of the writer ID, the content text, and PHOS [1] representation derived from the content text. These inputs collectively provide the necessary information for the network to generate the desired output representations, encoding the spatial distribution and relationships of characters and sub-characters within handwritten text images. By incorporating PHOS as a conditioning variable, the model gains an enhanced ability to capture and replicate the structural characteristics and style of the target domain, even for previously unseen character styles and distributions it was not explicitly trained. The PHOS technique plays a crucial role in the conditioning process of our diffusion model. PHOS extracts multi-scale histograms encoding the statistical distribution of shape and character features. By analyzing this hierarchical representation, our model captures fine-grained stylistic details and variations in handwriting. This enhances the diffusion model's ability to model the intricate stylistic nuances of handwriting, leading to more faithful style modeling compared to text conditioning alone. The PHOS representation is computed from the input word text and passed through an embedding layer to obtain a dense vector representation. This embedded PHOS vector is then concatenated with the embedded text representation and fed into the U-Net at each timestep, as illustrated in Fig. 3. The additional information provided by PHOS includes the geometric and spatial relationships between different character components, which are crucial for replicating the natural variation found in human handwriting. This joint conditioning on both text and PHOS enables our model to generate images that closely match the desired content and style, ensuring that the synthetic handwriting exhibits realistic variations and stylistic consistency. By incorporating PHOS, the model benefits from a richer, more detailed conditioning input that goes beyond simple text features, allowing it to more accurately capture the complexities of handwritten text generation. This integration significantly improves the model's performance in generating high-quality, stylized handwritten text. It also enhances the overall robustness of the generated images, making them suitable for a wider range of applications. Ultimately, PHOS-based conditioning sets a new standard in the field of synthetic handwriting generation.

The diffusion model is trained to estimate the noise $\epsilon_\theta(x_t, t)$, as shown in Eq. 3. The objective is to model the conditional probability $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, where $\mathbf{x}_{t-1}$ represents the clean latent representation at timestep $t-1$ given the current prediction $\mathbf{x}_t$. This probability distribution guides the iterative reverse sampling process, which aims to gradually refine the noisy image by removing noise at each timestep. To provide temporal information to the model, timesteps are integrated using a sinusoidal position embedding technique. This encoding method, inspired by [23], allows the model to differentiate and be aware of the specific timestep it is processing. During training, the model learns to denoise the corrupted latents by minimizing the reconstruction error between the predicted noise and the actual noise added to the latent, as Eq. 3 depicts the formulation of the diffusion loss function.

4.2 Sampling

We employ two different iterative sampling techniques for handwriting generation. The first approach is the conventional method that precisely follows the reverse denoising process across all 600 timesteps to yield high-fidelity samples $\mathbf{x}_{0:T}$ resembling the distribution $p_\theta(\mathbf{x}_{0:T})$. It takes a long time to generate data samples using this method as it involves sampling sequentially from the full conditional distribution $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ across all timesteps t. To handle this issue, we propose an early sampling technique that modifies the conventional method to predict noise values only every 5 timesteps, effectively sampling from an initial portion of $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ over only one-fifth of the original timesteps. This reduces the number of steps required per sample by a factor of 5x. The following subsection explains these methods in detail.

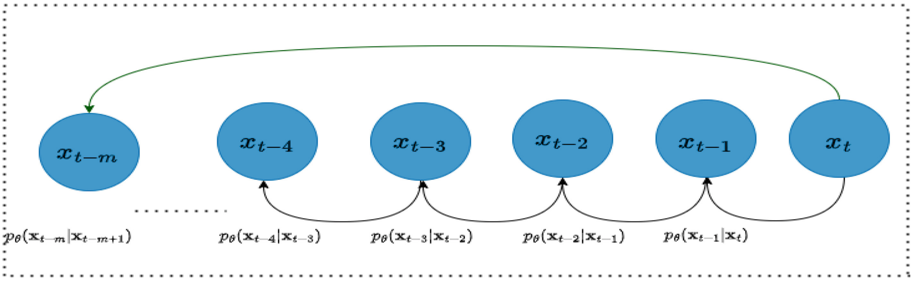

x_{t-m}
x_{t-4}
x_{t-3}
x_{t-2}
x_{t-1}
x_t
$p_\theta(\mathbf{x}_{t-m}|\mathbf{x}_{t-m+1})$
$p_\theta(\mathbf{x}_{t-4}|\mathbf{x}_{t-3})$
$p_\theta(\mathbf{x}_{t-3}|\mathbf{x}_{t-2})$
$p_\theta(\mathbf{x}_{t-2}|\mathbf{x}_{t-1})$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$

Fig. 4. Sampling steps

Iterative Reverse Sampling via the Complete Process To generate synthetic images resembling the training data distribution, we utilize the reverse denoising process. Specifically, we use the trained model to initiate the generation from Gaussian noise, unfolding the procedure across 600 timesteps as shown in Fig. 4. The reverse steps are shown by black arrows. At each step, the model removes a part of the noise from the noisy image produced in the previous step according to the probability distribution $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, where $\mathbf{x}_{t-1}$ represents the clean latent representation at timestep $t-1$ and $\mathbf{x}_t$ represents the noisy image at timestep t, which varies from 600 to 1.

This captures the model's ability to predict the clean latent representation $\mathbf{x}_{t-1}$ given the current prediction $\mathbf{x}_t$. The generation is guided using the same conditioning inputs: text, writer ID, and PHOS vector specific to image ID. This ensures the generated images not only depict the given text but also exhibit styling consistent with the training writer. Initially starting from a randomly noisy sample, denoising is sequentially performed at each timestep in reverse order, with the network predicting and subtracting the noise level guided by this probability distribution. This iterative process gradually reconstructs the clean latent representation by timestep 0. Finally, decoding this denoised latent using the decoder yields the synthetic pixel-space image resembling the style and content of the training data. In WordStylist [18], the same sampling procedure is employed to generate a new split of the training data, with the exception of the addition of the PHOS vector.

Generating Diffusion Samples Using Early sampling The conventional sampling technique utilized in WordStylist [18] involves a minimum of 600 to 1000 timesteps to generate each sample, sampling sequentially from the full conditional distribution $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$. However, when the goal is to generate many examples with varied writer IDs, this approach is prohibitively computationally expensive due to the large number of timesteps required per sample.

To address this challenge, we propose an efficient early-sampling technique. Rather than predicting noise using a trained diffusion model for each step we predict it after an interval of 5 steps, skipping the prediction for the remaining 4 steps in that interval, for these steps between intervals earlier predicted noise is used for denoised image estimate at step t. Below Eq. 4, represents image denoising.

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha}}\left(\mathbf{x}_t - \frac{1-\alpha}{\sqrt{1-\alpha}} \cdot \text{noise}_{\lfloor t/5 \rfloor \times 5}\right), \tag{4}$$

where $\text{noise}_{\lfloor t/5 \rfloor \times 5}$ denotes the noise term calculated at the last multiple of 5 not exceeding t.

Figure 4 summarises the procedure with only a green color arrow. The algorithm with the Pseudo-code given below follows a modified approach, where the prediction of noise is done only after every 5 steps. Prior to predicting the noise, the noise values remain constant from the previous prediction for the preceding 5 steps. The algorithm takes as inputs the trained diffusion model, the number of timesteps N (set to 600 for IAM and 1000 for CVL), and the value α (which is calculated as $1 - \beta$).

Pseudo-Code for Early-Sampling Technique. The algorithm begins by initializing the latent representation $\mathbf{x}$ as random noise. It then iterates through the timesteps from N to 1. Suppose the current timestep t is a multiple of 5 ($t \mod 5 = 0$). In that case, the algorithm predicts the noise $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ for that timestep using the trained diffusion model, The input to model is x_t which is conditioned on writer ID, text string, PHOS representation of the text string. The predicted noise for timestep t is denoted as $noise_t$. Next, the latent representation $\mathbf{x}_t$ is updated using the Eq. 4 to $\mathbf{x}_{t-1}$.

Algorithm 1: Early-Sampling Technique

```
Data: Trained diffusion model, number of timesteps $N = 600$, $\alpha = 1 - \beta$
Result: Final image $x_0$ as generated sample
$x \leftarrow$ random noise;
for $t \leftarrow N$ to 1 do
    if $t \mod 5 == 0$ then
        $noise_t \leftarrow$ prediction using diffusion model;
    $x_{t-1} \leftarrow \frac{1}{\sqrt{\alpha}}\left(x_t - \frac{(1-\alpha)}{\sqrt{1-\alpha^*}} \cdot noise_t\right)$;
return $x_0$;
```

4.3 Validation Framework for Data Generated Using Early Sampling

To ensure the quality and correctness of the synthetically generated images using the early sampling technique, we have implemented a robust validation framework. This framework incorporates a state-of-the-art Handwritten Text Recognition (HTR) model that assesses the generated images based on their textual content. It is important to note that this validation framework is particularly pertinent when the early sampling approach is used. By comparing the recognized text from the generated images with the intended text, the validation framework determines whether the images meet the expected standards. The validation process is an essential step in the synthetic data generation pipeline. It acts as a safeguard to maintain the integrity and reliability of the generated images. The HTR model employed in the validation framework is trained on the original IAM training split. During the validation phase, each diffusion-generated image undergoes a thorough evaluation. The recognition model processes the image and extracts the textual content. This recognized text is then compared against the ground truth text associated with the image. If the recognized text matches the ground truth, the generated image is considered correct and validated. Only the generated images that successfully pass the validation framework are included in the final synthetic dataset. This selective inclusion ensures that the synthetic data is of high quality and aligns with the expected textual content. By filtering out any generated images that do not meet the validation criteria, we maintain the integrity and usefulness of the synthetic data for downstream tasks such as training HTR models.

5 Experiment and Result Discussions

5.1 Experiment Setup

The evaluation of the proposed DiffWord framework involved two main categories of experiments. The first category focused on training generative diffusion models. The second category encompassed assessing the framework's performance in downstream tasks, such as HTR, writer classification, and calculating FID and KID scores. For all the experiments, we utilized the IAM Handwritten Dataset [16] and CVL [14] dataset, which are well-established benchmark datasets in the field of HTR. The dataset consists of a diverse collection of handwritten English text, including various writing styles and multiple writers. This dataset provides a suitable foundation for training our model to generate styled handwritten text images.

To ensure a fair comparison with WordStylist [18], we adopted the same IAM data split as that of WordStylist for diffusion training. For the CVL dataset, we used the officially provided split for train and test. It has around 87K train and 12K test words. The WordStylist model [18] is capable of generating only alphabetic characters. Consequently, when using both datasets [16] for downstream tasks, we preprocessed the official data split by discarding words that contain only non-alphabetic characters. For words that partially contain non-alphabetic characters, we replaced those characters with whitespace.

The diffusion model training follows a similar approach to [18]. We trained two separate models on the training set of the respective datasets, utilizing the text content and the corresponding PHOS representations and writer-ID as conditioning variables. The model was optimized using the same loss function and training hyperparameters as specified in WordStylist [18] until the loss value converged.

5.2 HTR Results

The Tables 1 and 2 compares the HTR accuracy achieved when training on datasets containing original training data augmented with synthetic samples generated by either the proposed DiffWord framework or the existing WordStylist [18] method. The evaluation is conducted on test data using the Character Error Rate (CER) and Word Error Rate (WER) metrics, where lower values indicate better performance. It is important to note that HTR framework is trained using a combination of original images from the training split of respective datasets and synthetic images generated by the same training image ID's by respective

Table 1. HTR comparison with Wordstylist on IAM dataset colour codes black: full denoising, green: early sampling, blue: the combined approach, pink: without Validation FrameWork

Generation Framework	**Dataset**	**No of images**	**CER % ↓**	**WER %↓**	**CER % ↓**	**WER %↓**
			with Validation		**without**	**Validation**
Real IAM	Original	44K			5.53	15.93
Wordstylist [18]	Original + **Synthetic**	86K			5.41	15.87
Wordstylist [18]	Original + **Synthetic**	86K	6.93	15.94	6.96	15.98
Wordstylist [18]	Original + **Synthetic**	128K	6.69	15.25		
DiffWord	Original + **Synthetic**	86K			4.39	12.89
DiffWord	Original + **Synthetic**	86K	5.15	14.83	5.33	14.98
DiffWord	Original + **Synthetic**	128K	4.38	12.83	5.98	13.80
DiffWord	Original + **Synthetic**	171K	4.36	12.81		
DiffWord	Original + **Synthetic**	214K	4.22	12.25		

frameworks. The term ‘Original’ in Table 1 and 2 corresponds to images from the train-split of the respective IAM or CVL dataset. On the other hand, ‘Synthetic’ represents images generated using the diffusion framework. In contrast to WordStylist [18], which exclusively employs the full denoising process described in Sect. 4.2 for image generation, DiffWord produces synthetic data through a flexible approach. It leverages either the full denoising process, early sampling, or a hybrid of both. Table 1 and 2 employs a color-coding scheme to delineate the methods used: black for full denoising, green for early sampling, blue for the combined approach, and pink to show without using validation framework. It is crucial to acknowledge that the validation framework is exclusively employed when synthetic data is produced via early sampling or the combined approach, specifically when data is generated with fewer steps. It is not utilized in the context of full denoising.

Table 2. HTR comparison with Wordstylist on CVL dataset. Colour codes black: full denoising, blue: the combined approach , pink: without Validation FrameWork

Generation Framework	**Dataset**	**No of images**	**CER % ↓**	**WER %↓**	**CER % ↓**	**WER %↓**
			with Validation		**without**	**Validation**
Real CVL	Original	86K			15.983	22.503
Wordstylist [18]	Original + **Synthetic**	86K			15.94	22.403
DiffWord	Original + **Synthetic**	164K			15.517	21.604
DiffWord	Original + **Synthetic**	250K	15.45	21.304	**15.491**	**21.47**
DiffWord	Original + **Synthetic**	339K	**14.98**	**21.28**	**15.13**	**21.293**

In Table 1, 1st row indicates the result on the original IAM data. The next 3 rows show a performance of the WordStylist [18]. 2nd row shows performance of the WordStylist [18] with CER 5.41% and WER 15.87%. DiffWord uses early sampling with validation and gets a CER of 5.15% and a WER of 14.8% (row 6) with the same number of samples of train data. This result is improved to a CER of 4.39% and a WER of 12.89% (row 5) when the complete denoising process is used but this process is difficult to scale up. The recognition accuracy consistently improves with the inclusion of additional synthetic samples produced by the proposed scalable early sampling technique, as demonstrated in rows 7, 8, and 9 of columns 4 and 5 in Table 1. This demonstrates that the images produced via early sampling are as effective as those from the complete denoising process for improving performance when augmenting training data. A similar pattern is also observed in Table 2. Where adding the augmented data with the original improves HTR performance and DiffWord framework outperforms WordStylist [18]. The consistent accuracy gains highlight DiffWord's ability to harness useful information from both generation methods to further reduce error rates with increased data.

The Tables 1 and 2 reveal that the DiffWord framework consistently outperforms, demonstrating improved recognition accuracy. The validation framework, while important for ensuring the quality and correctness of the generated images, plays a secondary role compared to the main contributions of the early sampling algorithm and the PHOS representation. The results in columns 4 and 5 of Table 1 and 2, which include the validation framework, show marginal improvements over those without it (highlighted in pink), underscoring its supportive role. The relevant rows for Table 1 are 6,7 and for 2 are 4,5. Where we can compare CER and WER with and without validation. However, the principal improvements in recognition accuracy are attributed to the effective combination of early sampling and PHOS representation, as evidenced by the significant gains in CER and WER metrics. Overall, the early sampling technique and PHOS representation are the critical components driving the superior performance of the DiffWord framework. The validation framework enhances the reliability of the generated data, ensuring it meets the expected standards, but the core advancements stem from the primary methodologies employed. The experimental results validate that integrating these techniques leads to more accurate and effective handwriting recognition systems.

5.3 HTR Training on OOV Data

In this section, we explore the capabilities of the diffusion model in generating OOV data, and the subsequent training of a HTR model on this data. The focus lies on contrasting the proficiency of DiffWord with that of the WordStylist [18] model in producing OOV handwritten words. Figure 5 compares the handwritten words generated with the same text and writer ID. It is qualitatively clear that in the DiffWord-generated handwritten images, the overall handwriting quality appears more refined, exhibiting smoother and more consistent letterforms from the Fig. 5. This consistency contributes to enhanced readability and reduces potential ambiguities in character recognition. It is clear that DiffWord excels in capturing the fine details of handwriting, including subtle variations in stroke thickness, letter spacing, and individual character shapes. These precise details contribute to a more authentic and realistic representation of handwritten text.

Table 3 presents the outcomes of the HTR model's learning efficacy when trained on the OOV data crafted by both diffusion models trained on the IAM split. To generate OOV data, we considered the set of 1736 OOV word classes(those word classes are not present in the IAM dataset) and generated OOV class images by using all the writers present in the IAM dataset and each generating 5 samples of a particular word class. This generated data is then passed through a five-fold validation framework The test split in each fold was carefully curated to ensure that if a writer's style appeared in the test set for a particular word, that writer was not included for that same word in the respective training split of that fold. This approach simulates a more challenging and realistic scenario where the model is tested on completely unseen handwriting styles. By repeating this process across all splits, we aim to provide a robust

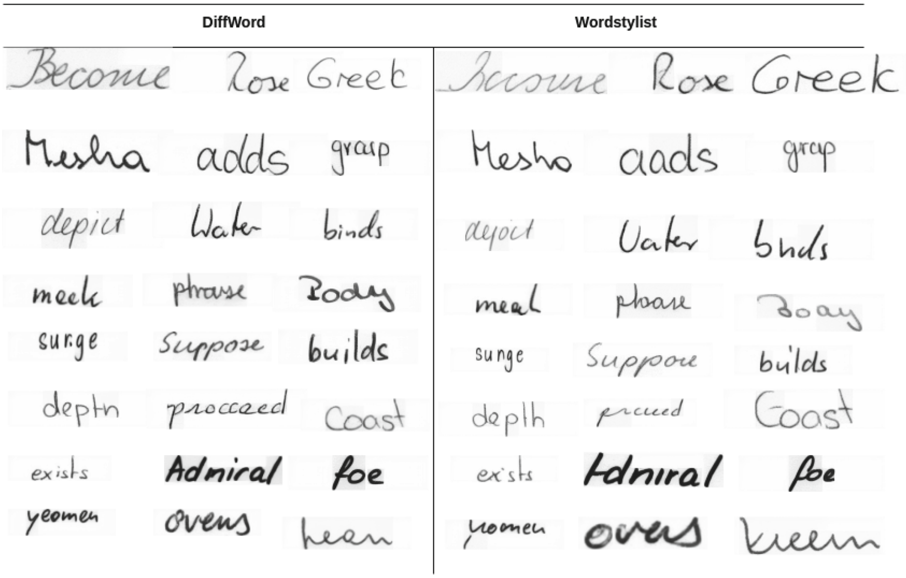


Fig. 5. Out-of-Vocabulary Comparison

Table 3. CER and WER scores on OOV data for DiffWord and WordStylist using a Five-fold validation framework

Fold	DiffWord		WordStylist	
	CER (%)	WER (%)	CER (%)	WER (%)
1	2.9	6.7	3.8	8.1
2	2.8	6.3	3.41	7.4
3	2.8	6.4	3.4	7.43
4	2.8	6.9	3.5	7.4
5	2.5	5.6	3.1	6.76

assessment of the model's ability to generalize to writers and validate its performance consistency.

To further scrutinize our model's performance on OOV words, we segmented the analysis based on the word length of the above-generated OOV data. Our objective was to determine how the DiffWord model's proficiency varies with the length of the words it generates. We divided the words into three bins: words with lengths from 0 to 3 characters, from 4 to 6 characters, and more than 6 characters. The analysis provides insight into the model's effectiveness in dealing with short, medium, and long words, which are often differently challenged by the intricacies of handwriting styles. For each bin, we computed the WER, The results are shown in the Table 4 The table shows the WER by word length bins across 5 folds of data, with the number of samples in each bin indicated in parentheses.

5.4 Writer Classification

Writer classification is essential for understanding and validating the effectiveness of synthetically generated handwritten text images that are not only realistic but also stylistically accurate. It serves as a means to assess whether the generated synthetic data maintains the unique styles of different writers, which is crucial for applications such as authorship verification, forensic analysis, and personalized digital content creation. To evaluate the style preservation capability of our proposed DiffWord framework, we conducted experiments on word-level writer classification. For this, the original training splits of both datasets (IAM and CVL) are used as training data, and separate writer identification models are created for IAM and CVL data. We synthetically generated train split images of both datasets using conventional full-sampling. This generated data is used as test data for writer classification. If the synthetic data matches the style distribution of the original data, then it will be reflected in test accuracy. In our approach, we employ a Convolutional Neural Network (CNN) with pre-trained

Table 4. Word Error Rate (WER) with respect to Word Length, numbers inside bracket denotes the number of samples.

Word Length	WER% Fold 1	WER% Fold 2	WER% Fold 3	WER% Fold 4	WER% Fold 5
<3	7(114)	7 (120)	3 (120)	14(110)	21(75)
4–6	6(64270)	6 (64260)	6 (62945)	6(60135)	6(47545)
>6	7(11516)	7 (11520)	7 (11315)	6(10575)	5(7785)
Overall	**6.7**	**6.3**	**6.4**	**6.9**	**5.6**

Table 5. Word level Writer Classification Evaluation Results

Method Name	Test Data Accuracy %	CVL Data Accuracy % ↑
DiffWord	**66.5**	**45.1**
WordStylist [18]	63.5	37.3

ResNet18 [3] as a backbone for writer classification. This ResNet18 is initially pre-trained on the ImageNet dataset [5] and subsequently fine-tuned using the respective datasets, specifically tailored for the task of writer classification at the word level. Notably, the ResNet18 architecture used in our approach is the same as the one utilized in [18].

The comparison of DiffWord with wordStylist [18] are as shown in Table 5. Our proposed DiffWord method achieves better word-level writer classification accuracy compared to the WordStylist on both datasets. We believe this is due to DiffWord's use of PHOS representations as an additional conditioning input, which captures richer structural and spatial information about the writing style characteristics compared to WordStylist [18] which only conditions on text.

5.5 Comparative Analysis of Image Quality

The Fréchet Inception Distance (FID) and Kernel Inception Distance (KID) are both crucial metrics used for evaluating the quality of images generated by generative models. FID measures the similarity between the distributions of generated images and real images, focusing on capturing both the texture and feature distribution, where lower scores indicate better quality and higher resemblance to the real dataset. In conjunction with FID, KID provides an alternative measure by evaluating the similarity between the feature distributions of real and generated images using the kernel trick. This metric offers an unbiased estimate of similarity and is particularly sensitive to mode collapse in generative models. Lower KID values suggest a closer match between the feature distributions of generated and real images, indicating higher quality of the generated outputs. Both metrics together provide a comprehensive assessment of the quality of generated images, with FID focusing on closeness to real image features and KID assessing the distributional similarity in a statistically robust way.

Table 6 shows the comparison of both FID and KID scores for the generated data. From the results, it is clear that DiffWord outperforms Wordstylist [18]. A lower FID score, as achieved by the DiffWord method, indicates that this model generates images that have features more similar to the real dataset than those generated by the Wordstylist model. Moreover, the lower KID score achieved by DiffWord reinforces this outcome by suggesting that the distribution of the generated images is closer to that of the real images. This dual superiority in FID and KID scores demonstrates that DiffWord can create more realistic

Table 6. FID and KID Comparison for IAM and CVL Datasets

Dataset	Method Name	FID ↓	KID ↓
IAM	**DiffWord**	67	0.0118
	Wordstylist [18]	90	0.0175
CVL	**DiffWord**	14.0916	0.0059
	Wordstylist [18]	14.1391	0.0060

handwritten text images, potentially reducing the likelihood of mode collapse and ensuring a diverse output that closely mirrors the true data distribution.

5.6 Potential Limitations and Challenges

The diffusion process and PHOS conditioning increase computational complexity, leading to longer training times and higher resource requirements. Balancing efficient computation with high-quality handwriting generation remains a critical challenge. Additionally, it is vital that synthetic data accurately captures the variability of real handwriting to be effective in downstream tasks. To address this, we have implemented a validation framework. However, the generation of synthetic handwriting data also raises ethical and privacy concerns, making it essential to ensure that the generated data does not inadvertently reproduce sensitive information.

6 Conclusion

Our research introduces WordDiff novel method for handwritten word image generation, which outperforms existing WordStylist [18] model in generating both in-vocabulary and out-of-vocabulary words, as evidenced by superior image quality assessment and writer classification results. The integration of PHOS into the conditioning process allows our model to more accurately capture the distinct handwriting styles of individual writers, leading to more realistic and stylistically consistent text generation. Furthermore, the early sampling technique we introduced helps in reducing the computational resources and time required for data generation, without compromising on quality. These improvements are quantitatively supported by enhanced recognition and writer classification accuracy, FID scores, indicating higher quality of the generated images. While our model demonstrates improvements over existing methods, it represents a promising step towards enriching synthetic data resources and potentially aiding in the refinement of HTR systems.

Acknowledgement. We are grateful to Simula HPC cluster, The research presented in this paper has benefited from the Experimental Infrastructure for Exploration of Exascale Computing (eX3), which the Research Council of Norway financially supports under contract 270053. This work is supported under the aegis of the Hugin-Munin project funded by the Norwegian Research Council (Project number 328598).

References

1. Bhatt, R., Rai, A., Chanda, S., Krishnan, N.C.: Pho(SC)-CTC—a hybrid approach towards zero-shot word image recognition. Int. J. Doc. Anal. Recogn. (IJDAR) (2022)
2. Bhunia, A.K., Khan, S.H., Cholakkal, H., Anwer, R.M., Khan, F.S., Shah, M.: Handwriting transformers. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1066–1074 (2021). https://api.semanticscholar.org/CorpusID:233181822

3. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. ArXiv **abs/1809.11096** (2018). https://api.semanticscholar.org/CorpusID:52889459
4. Davis, B.L., Tensmeyer, C., Price, B.L., Wigington, C., Morse, B., Jain, R.: Text and style conditioned gan for the generation of offline-handwriting lines. ArXiv **abs/2009.00678** (2020). https://api.semanticscholar.org/CorpusID:221448289
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009). https://api.semanticscholar.org/CorpusID:57246310
6. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. ArXiv **abs/2105.05233** (2021). https://api.semanticscholar.org/CorpusID:234357997
7. Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., Litman, R.: Scrabblegan: semi-supervised varying length handwritten text generation. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4323–4332 (2020). https://api.semanticscholar.org/CorpusID:214623091
8. Gan, J., Wang, W.: Higan: handwriting imitation conditioned on arbitrary-length texts and disentangled styles. In: AAAI Conference on Artificial Intelligence (2021). https://api.semanticscholar.org/CorpusID:235306524
9. Gan, J., Wang, W., Leng, J., Gao, X.: Higan+: handwriting imitation gan with disentangled representations. ACM Trans. Graph. **42**(1) (2022). https://doi.org/10.1145/3550070
10. Graves, A.: Generating sequences with recurrent neural networks. ArXiv **abs/1308.0850** (2013). https://api.semanticscholar.org/CorpusID:1697424
11. Ha, D.R., Eck, D.: A neural representation of sketch drawings. ArXiv **abs/1704.03477** (2017). https://api.semanticscholar.org/CorpusID:8968704
12. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. ArXiv **abs/2006.11239** (2020). https://api.semanticscholar.org/CorpusID:219955663
13. Kang, L., Riba, P., Wang, Y., Rusiñol, M., Forn'es, A., Villegas, M.: Ganwriting: content-conditioned generation of styled handwritten word images. ArXiv **abs/2003.02567** (2020). https://api.semanticscholar.org/CorpusID:212414769
14. Kleber, F., Fiel, S., Diem, M., Sablatnig, R.: Cvl-database: an off-line database for writer retrieval, writer identification and word spotting. In: 2013 12th International Conference on Document Analysis and Recognition, pp. 560–564 (2013). https://doi.org/10.1109/ICDAR.2013.117
15. Luo, C., Zhu, Y., Jin, L., Li, Z., Peng, D.: Slogan: handwriting style synthesis for arbitrary-length and out-of-vocabulary text. IEEE Trans. Neural Netw. Learn. Syst. **34**, 8503–8515 (2022). https://api.semanticscholar.org/CorpusID:247058493
16. Marti, U.V., Bunke, H.: The iam-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recogn. **5**, 39–46 (2002). https://api.semanticscholar.org/CorpusID:29622813
17. Mattick, A., Mayr, M., Seuret, M., Maier, A.K., Christlein, V.: Smartpatch: improving handwritten word imitation with patch discriminators. ArXiv **abs/2105.10528** (2021). https://api.semanticscholar.org/CorpusID:235166331
18. Nikolaidou, K., et al.: Wordstylist: styled verbatim handwritten text generation withÂ latent diffusion models. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) Document Analysis and Recognition - ICDAR 2023, pp. 384–401. Springer, Cham (2023)
19. Pippi, V., Cascianelli, S., Cucchiara, R.: Handwritten text generation from visual archetypes. In: 2023 IEEE/CVF Conference on Computer Vision and Pattern

Recognition (CVPR), pp. 22458–22467 (2023). https://api.semanticscholar.org/CorpusID:257766680

20. Retsinas, G., Sfikas, G., Gatos, B., Nikou, C.: Best practices for a handwritten text recognition system. In: International Workshop on Document Analysis Systems (2022). https://api.semanticscholar.org/CorpusID:248949489
21. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. ArXiv **abs/1505.04597** (2015). https://api.semanticscholar.org/CorpusID:3719281
22. Sohl-Dickstein, J.N., Weiss, E.A., Maheswaranathan, N., Ganguli, S.: Deep unsupervised learning using nonequilibrium thermodynamics. ArXiv **abs/1503.03585** (2015). https://api.semanticscholar.org/CorpusID:14888175
23. Vaswani, A., et al.: Attention is all you need. In: Neural Information Processing Systems (2017). https://api.semanticscholar.org/CorpusID:13756489
24. Vögtlin, L., Drazyk, M., Pondenkandath, V., Alberti, M., Ingold, R.: Generating synthetic handwritten historical documents with OCR constrained GANs. ArXiv **abs/2103.08236** (2021). https://api.semanticscholar.org/CorpusID:232233484
25. Yang, Y., Gui, D., Yuan, Y., Ding, H., Hu, H.R., Chen, K.: Glyphcontrol: glyph conditional control for visual text generation. ArXiv **abs/2305.18259** (2023). https://api.semanticscholar.org/CorpusID:258960586
26. Zdenek, J., Nakayama, H.: Jokergan: memory-efficient model for handwritten text generation with text line awareness. In: Proceedings of the 29th ACM International Conference on Multimedia (2021). https://api.semanticscholar.org/CorpusID:239011850
27. Zhang, X.Y., Yin, F., Zhang, Y., Liu, C.L., Bengio, Y.: Drawing and recognizing Chinese characters with recurrent neural network. IEEE Trans. Pattern Anal. Mach. Intell. **40**, 849–862 (2016). https://api.semanticscholar.org/CorpusID:3708198
28. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2242–2251 (2017). https://api.semanticscholar.org/CorpusID:206770979

KaiRacters: Character-Level-Based Writer Retrieval for Greek Papyri

Marco Peer[1](✉), Robert Sablatnig[1], Olga Serbaeva[2], and Isabelle Marthot-Santaniello[2]

[1] Computer Vision Lab, TU Wien, Austria
mpeer@cvl.tuwien.ac.at
[2] Departement Altertumswissenschaften, Universität Basel, Basel, Switzerland

Abstract. This paper presents a character-based approach for enhancing writer retrieval performance in the context of Greek papyri. Our contribution lies in introducing character-level annotations for frequently used characters, in our case the trigram *kai* and four additional letters ($\varepsilon, \kappa, \mu, \omega$), in Greek texts. We use a state-of-the-art writer retrieval approach based on NetVLAD and compare a character-level-based feature aggregation method against the current default baseline of using small patches located at SIFT keypoint locations for building the page descriptors. We demonstrate that by using only about 15 characters per page, we are able to boost the performance up to 4% mAP (a relative improvement of 11%) on the GRK-120 dataset. Additionally, our qualitative analysis offers insights into the similarity scores of SIFT patches and specific characters. We publish the dataset with character-level annotations, including a quality label and our binarized images for further research.

Keywords: Greek Papyri · Writer Retrieval · Historical Documents · Document Analysis

1 Introduction

Writer Retrieval (WR) describes the task of finding documents penned by the same writer as a given query document. Typical applications include forensics or digital humanities, in particular WR for historical documents, which also includes papyrology, the study of ancient texts on papyri [9]. Scholars face difficulties in assigning writers due to degradation in papyri and variations in handwriting over time. By tracing handwriting evolutions based on solid evidence, WR is a possibility to organize papyrological documentation coherently, that is then used for further tasks, e.g., reconstructing ancient archives, refining the accuracy of writer identification or papyri dating [5]. To aid papyrologists through computerized and automated WR, deep-learning-based methods have emerged as a promising technique in the field of papyrology [3–5,16,17].

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 73–88, 2025.
https://doi.org/10.1007/978-3-031-78495-8_5

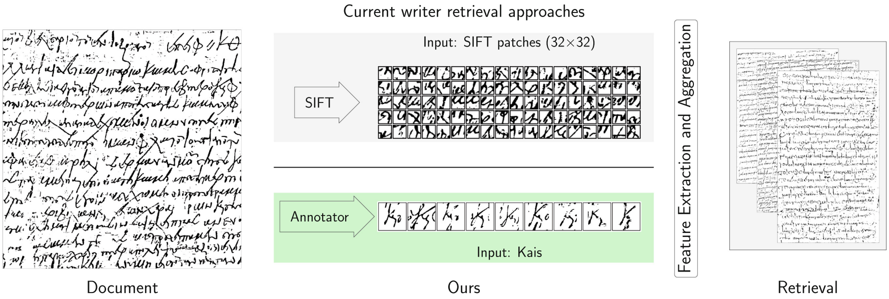

Current writer retrieval approaches
Input: SIFT patches (32×32)
SIFT
Annotator
Input: Kais
Feature Extraction and Aggregation
Document
Ours
Retrieval

Fig. 1. Overview of our approach. Contrary to state of the art, we do not aggregate deep features of SIFT patches that usually contain a few strokes of handwriting. Instead, we aggregate features of specific characters, in our case the trigram *kai*, to form the global page descriptor.

WR methods usually rely on a multi-stage approach: *Sampling* patches of handwriting (e.g. via SIFT keypoint detection [2,15]) combined with preprocessing, such as binarization, feature extraction via a neural network and an encoding stage where the statistics of the features are calculated (e.g., Vector of Locally Aggregated Descriptors (VLAD) [2] or NetVLAD [14,15,19]) and aggregated. In our paper, we want to extend the current state of the art and investigate the sampling and aggregation step by using character-level annotations of the handwriting to aggregate the NetVLAD-encoded features of specific characters. More specifically, we keep the unsupervised training step, where we cluster SIFT descriptors and use the cluster assignment as a target label of the 32×32 patch extracted at the keypoint, but use only features of specific characters for aggregation during inference for the global page descriptor. An overview of our methodology is given in Fig. 1. While current approaches rely on thousands of local features, extracted by a neural network at SIFT keypoint locations (which we refer to as *SIFT patches*), we investigate the aggregation of the features of specific characters, in our case *kai*, a trigram usually standing for the most frequent word in Greek texts, the conjunction meaning "and". It is also used by scholars to determine authorship of Greek texts [7,10]. In our work, we concentrate on Greek papyri since this domain still lacks performance for WR or writer identification [3,16], with Mean Average Precision (mAP) of only about 40%. Additionally to our evaluation, we publish the dataset[1] of the full images of the GRK-120 dataset - that was until now only available as segmented rows in [5] along with the annotated characters (bounding boxes and character images in color and binarized) - and we compare performances for other characters (epsilon ε, kappa κ, mu μ, and omega ω).

We show that by using character-level annotations and aggregating only specific characters, we 1) boost the retrieval performance in terms of mAP, even when the quality of the letters is low, and 2) reduce the amount of handwriting

[1] Dataset: https://d-scribes.philhist.unibas.ch/en/case-studies/dioscorus/kairacters/.

needed to obtain a discriminate global page descriptor (11 *kai*-s per page vs. 2600 SIFT patches). Our evaluation is thoroughly conducted on subsets of the GRK-120 dataset, ensuring fair comparisons on the WR performance. Furthermore, our qualitative studies show that the use of characters detects complementary similarities of the writers, rather than only improving it.

To summarize, our contributions are:

- We release our dataset - the full images as well as the images and annotations of all of our characters used and of the remaining 20 letters - in color and binarized version. For the *kai*-s, quality labels are included.
- We evaluate character-level-based feature aggregation with a state-of-the-art WR approach [15] and show that this outperforms the currently dominating methodology of aggregating features of SIFT patches.
- Our approach does only need a few samples to achieve similar performances as the baseline using SIFT patches. With only 11 *kai*-s, the performance is on par or even better compared to 2.6k SIFT patches per sample.
- We qualitatively evaluate the similarity of SIFT patches as well as the *kai*-s, providing insights for scholars and further research.

Our paper is structured as follows: Sect. 2 describes related work in the field of WR for Greek Papyri. In Sect. 3, details on the data used and the WR approach are given. The evaluation protocol is described in Sect. 4, and our results are presented in Sect. 5. We conclude our paper in Sect. 6.

2 Related Work

In the following, we provide a brief overview of previous work on WR and writer identification in the domain of papyri.

Pirrone et al. [17] explore a self-supervised method for retrieving papyri fragments utilizing a Siamese network with contrastive loss. Their evaluation includes the Michigan Papyrus Collection and a subset of HisFragIR20. Similar to our work, Christlein et al. [3] implement a previously established algorithm [1], training a network on clustered SIFT descriptors with the initial, 50 image-large GRK-Papyri dataset [9]. They rely on a U-Net-based binarization technique for papyri and demonstrate that eliminating degradation and background artifacts notably enhances the retrieval performance. The GRK-Papyri authors present baseline outcomes utilizing local NBNN [8], a learning-free algorithm based on SIFT descriptors, but struggle with document degradation issues. Nasir et al. [12] concentrate on binarization via Deep Otsu and train a neural network for writer identification using 512 × 512 patches. In [5], Cilia et al. introduced PapyRow, an extension of GRK-Papyri from 50 to 120 images released as fragments via line segmentation. In our previous work, we proposed a feature mixing network for the retrieval and identification of fragments [16], and show that the learned descriptors still rely on patterns included in the background. Another work of Cilia et al. [4] focuses on the writer identification of patches on a slightly extended dataset compared to the initial GRK-papyri (of 50 images from 10 writers to which they added one writer, Dios, and its 15 writing samples from GRK-120) with different convolutional backbones. In this work, our WR approach relies on a similar binarization technique as Christlein et al. [3] who use a U-Net and a manually designed augmentation pipeline for degradation usually found in papyri. Regarding the deep feature extraction of patches, we use a residual network and NetRVLAD as proposed in our previous work [15].

Greek papyri have been also involved in two recent competitions: the binarization competition DIBCO2019 [18] and the competition organized by Seuret et al. [21] at ICDAR2023, with the latter one primarily focusing on the localization and classification of characters, resulting in promising recognition outcomes. Our paper's contributions build upon those results. Although the characters we used were manually annotated by expert papyrologists, with the advance in character detection and classification in Greek papyri, our method can in the future be applied to further enhance WR performance and eliminate the need for manual annotation by scholars. Another recent task on Greek papyri is their chronological attribution, or estimating their date, where, e.g., Pavlopoulos et al. [13] apply a method based on Convolutional Neural Networks (CNNs).

3 Methodology

In this section, we describe the main parts of our methodology. We start with the description of the data, in particular the statistics including the characters, followed by the WR pipeline.

3.1 Data

The dataset used in this paper is based on the GRK-120 dataset, consisting of 120 documents belonging to 23 different writers that have been identified so far in the course of the D-Scribes[2] project. One of the aims of this project is the computer-assisted identification of scribes of the archive of Dioscorus of Aphrodito, the richest papyrus archive of the Byzantine period [9]. Secondly, the characters of the texts are annotated as two distinct subsets - 1300 *kai*-s and 9511 individual characters representing one of the 24 letters of the Greek alphabet. For our work described below, we only use part of it, but make the full data available for further research.

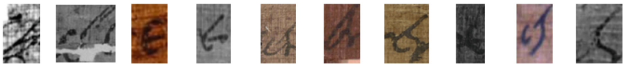
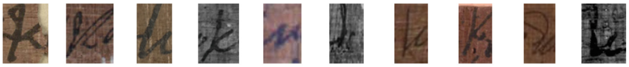
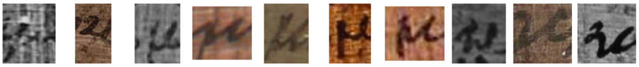
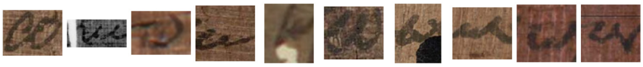
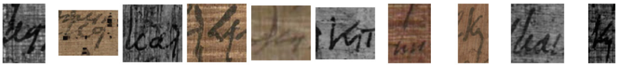
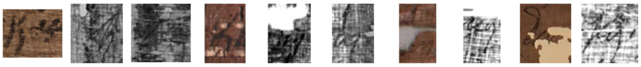
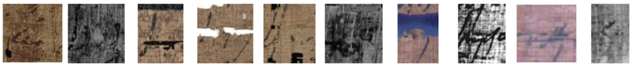
Annotations. The annotations of the characters of the texts in GRK-120 are done with the READ software[3]. We follow the workflow described in our previous work [20] and define two different sets of annotations, indicated in Table 1, which are publicly available:

- The first set consists of letters, usually of the best-preserved three lines of text. Since the annotations are not entirely complete, we provide only preliminary

[2] https://d-scribes.philhist.unibas.ch/en/.

[3] https://github.com/readsoftware/read.

Table 1. Statistics of the annotations for the characters and *kai*-s (BT`x` describes the quality label). We also show randomly sampled examples of the annotated characters.

Character		Samples	Examples
ε		504	
κ		353	
μ		287	
ω		318	
kai	BT1	720	
	BT2	380	
	BT3	51	

results using four of those characters (ε, κ, μ, ω). They are chosen by us since they are assumed to be discriminative for the handwriting of the scribes [6].

- The second set includes only *kai*-s, a trigram formed of kappa κ, alpha α and iota ι that in most of the cases stands for the most frequent word in the Greek language (corresponding to the English "and") but occasionally also occurs as part of a word (for instance *dikaios*, "just, fair"). This trigram was chosen not only because of its very high frequency that makes it likely to appear even in small papyri, but also because its shape is usually not affected by the previous or following character (no ligature before κ nor after ι). The quality label of each *kai* is additionally provided.

Quality Labeling. The *kai*-s are tagged according to their preservation state. We provide three labels (`BT{1,2,3}`), where `BT1` indicates best quality (no degradation), annotations tagged with `BT2` are partly damaged, and unreadable *kai*-s are assigned with `BT3`, where the *kai* is only identifiable given the context of the text. Examples and the quantity of each label are given in Table 1.

Final Dataset. The final dataset consists of *kai*-s and the characters ε, κ, μ and ω, and those are assigned to a writing sample (an image containing only one writer's handwriting) from the GRK-120 dataset. Note that both set of annotations are independent, e.g., the κ-s used for aggregation might be part of the *kai*-s. This

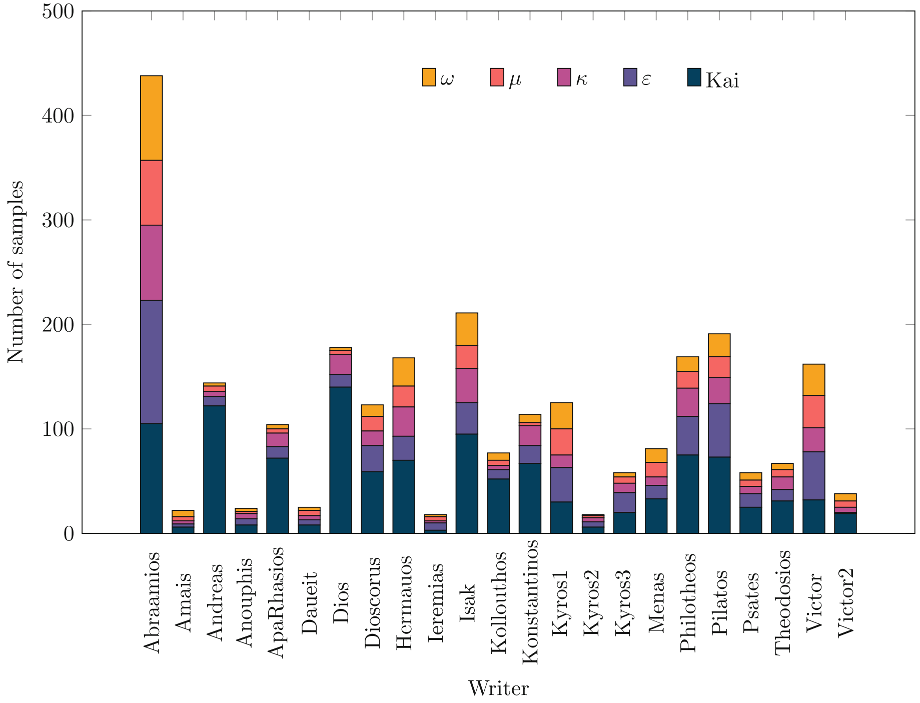


Fig. 2. Distribution of the characters used per writer

assignment allows us to compare the traditional WR approaches to our proposed character-based aggregation scheme. We provide *kai*-s for each of the 23 writers in the dataset. Since the number of pages the writers contributed in the GRK-120 dataset are highly imbalanced (e.g. Abraamios has 21 writing samples from 18 papyri), we also observe an uneven distribution of characters per writer, which increases the difficulty of WR by having only a small amount of handwriting for some writers, e.g. Kyros1 or Victor2. The distribution of the characters is also shown in Fig. 2. Due to the presence in the GRK-120 dataset of short and damaged writing samples, and a limited time for manual annotation, we are not able to provide *kai*-s or letters for each document. We provide more details on the subsets we evaluate our approach on in Sect. 4.

3.2 Writer Retrieval

In this section, we describe our method for WR. We start with preprocessing, which mainly consists of binarization, followed by the feature extraction and aggregation part.

Binarization. Given that our method is tailored for binarized handwriting and that Greek papyri typically exhibit severe degradation, such as artifacts, holes,

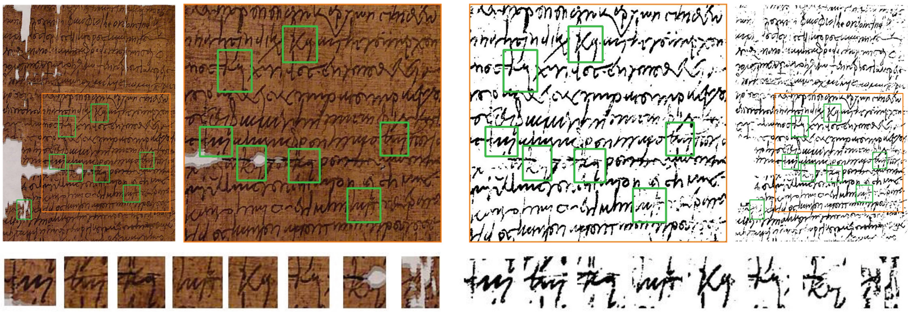


Fig. 3. Example page with annotated *kai*-s (green) in color and binarized. Note that we use a separate U-Net for binarizing characters and full pages (ID: Isak_5). (Color figure online)

or background patterns, we employ binarization as a preprocessing step to isolate the handwriting. Following the approach of Christlein et al. [3], we utilize a U-Net-based binarization technique, which has been demonstrated to outperform traditional methods for papyri. To address the binarization of full pages as well as crops of the annotated characters - parts of the characters might suffer from poor binarization, e.g. as shown in Fig. 3 - we apply two training strategies, both trained on grayscale images from the DIBCO2019 dataset [18], subset II that contains ten papyri:

1. For full pages, the binarized output may contain entirely white regions without any handwriting. We train a U-Net on randomly sampled 128 × 128 patches.
2. For images of characters, we assume accurate annotations and, therefore, no empty patches. Consequently, we focus our patch sampling exclusively on regions containing handwriting. Moreover, we train on patch sizes ranging from 32 to 128.

Examples of the two binarization methods, as well as the binarized *kai*-s, are shown in Fig. 3. Both networks are able to segment the handwriting and remove most of the artifacts from the papyri.

Feature Extraction. The retrieval method we use is based on the work proposed in [15]. It consists of a ResNet20 for feature extraction and NetRVLAD to encode the features. The network is trained in an unsupervised manner by a two-step approach: SIFT keypoints are detected, and the corresponding descriptors are clustered. We then use the cluster assignment as a surrogate label for training, where a 32 × 32 patch serves as input to the network.

Aggregation. We calculate the global page descriptor $\boldsymbol{X}$, of a page consisting of N samples of the NetRVLAD-encoded set of features $\mathcal{F} = \{\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{N-1}\}$ by sum-pooling $\boldsymbol{X} = \sum_{i=0}^{N-1} \boldsymbol{x}_i$ followed by element-wise power normalization

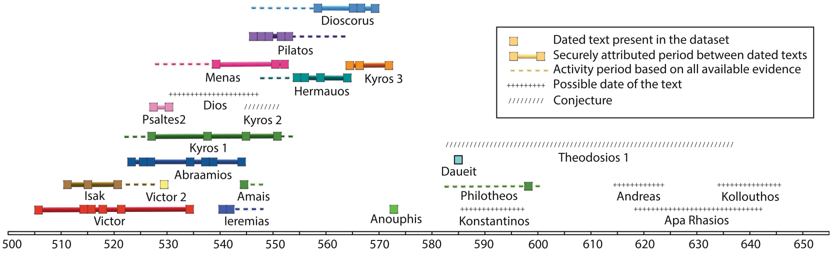


Fig. 4. Timeline of the documents included in GRK-120. The time of origin of some documents is still unknown.

$f(x) = \text{sign}(x)|x|^{\alpha}$ with $\alpha = 0.4$. Finally, we whiten the page descriptors and l_2-normalize again.

The key idea of our paper is that instead of following state-of-the-art approaches and performing inference using 32 × 32 patches extracted at SIFT keypoints, we aggregate a feature set $\mathcal{F}$ of previously defined characters for WR. Although our work relies on manual annotation, automated approaches, e.g., character classification, are an active field of research [21] and could be integrated into our pipeline to eliminate the need for human annotations.

In contrast to SIFT patches, which only contain a few strokes, we investigate the use of larger parts of handwriting, such as characters or trigrams (*kai*-s), since our network is not restricted to a specific image sizes. Since the annotated characters are usually larger than the strokes in the SIFT patches, we use a slightly higher image size (64 × 64) for the characters, compared to size 32 × 32 of the SIFT patches. We also conducted experiments for SIFT patches using higher image sizes, but this did not yield significant differences in performance.

4 Evaluation

Dataset. The GRK-120[4] dataset consists of 120 writing samples penned by 23 writers, with the number of documents per writer being highly unbalanced. Since the documents are contracts, the authorship of these samples is verified by experts' examination of scribes' signatures. Furthermore, the documents vary regarding their date, a timeline is shown in Fig. 4. The corpus dates from the 6th and 7th century CE, where the date of some papyri, e.g. written by Theodosius or Kollouthos, is still only an estimate.

We report our results on the GRK-120 dataset and, due to the limited availability of letters on specific pages, subsets of GRK-120, as shown in Table 2. We use the GRK-69 subset for the evaluation of combinations of letters and the GRK-110 version to evaluate the *kai*-s. *Kai-s* are more numerous because they have been carefully looked for and exhaustively annotated .

[4] https://d-scribes.philhist.unibas.ch/en/case-studies/dioscorus/.

Table 2. Variants of GRK-Papyri dataset used in our evaluation.

Set	Writers	
GRK-50	10	Baseline dataset for comparison with state of the art [9]
GRK-69	23	Evaluation of characters ε, μ, κ, ω
GRK-110	23	Evaluation of *kai*-s
GRK-120	23	Full dataset, extension of GRK-50

Training. We mainly follow the approach in [15]: We use ResNet20 and append NetRVLAD with 32 clusters as a feature extractor. Our networks are trained in an unsupervised manner by applying Cl-S [1] with 5000 clusters, hence we do not need any writer labels. For training, we use Adam optimizer for 30 epochs with a learning rate of 10^{-4} and a batch size of 1024.

Evaluation Protocol. Our approach is evaluated by using leave-one-image-out cross validation. Each document of the dataset is once used as a query, and the remaining documents are ranked according to the cosine similarity of the page descriptors. Our main metric used is mAP that considers the full ranked list. Furthermore, similar to [3], we report (soft) Top-1, Top-5, Top-10 scores. Top-x indicates if at least one document written by the same writer is included in the first x documents of the ranked list. Finally, we also calculate the precision at k (Pr@k), which describes the ratio of correct documents in the first k elements. All of our results are reported on an average of five runs with different seeds.

5 Results

In this section, we provide our main results. First, we start by comparing our method to state of the art, and follow with a description of results when using only predefined characters for WR. Finally, we also show qualitative results of the retrieval.

5.1 Baseline Performance

GRK-50. Firstly, to show the effectiveness of our method, we provide results when aggregating SIFT patches (referred to as baseline) on the GRK-50 [9] dataset. The state-of-the-art method is proposed by Christlein et al. [3], using Cl-S training as well as SIFT descriptors, both encoded by mVLAD. As shown in Table 3, our approach has a slightly lower mAP, but outperforms [3] regarding Top-1 accuracy. Additionally, we check if training on more data helps our network by training on the GRK-120 dataset. Interestingly, we observe a drop in performance on each metric reported which might be a result of complexity added to the training process, e.g. an increased amount of variety in handwriting by including more writers.

Table 3. GRK-50: Comparison to state of the art.

Method	Top-1	Top-5	Top-10	mAP
Cl-S + ResNet20 + NetRVLAD	**58.0**	74.0	**94.0**	39.1
Cl-S + ResNet20 + NetRVLAD (trained on GRK-120)	52.0	74.0	90.0	38.2
R-SIFT + mVLAD [3]	48.0	**84.0**	92.0	**42.8**
Cl-S + ResNet20 + mVLAD [3]	52.0	82.0	**94.0**	42.2

GRK-120 and Subsets. Next, we report the performance of the baseline on the subsets we subsequently use to evaluate our character-based approach. Results are presented in Table 4. The GRK-69 dataset achieves the lowest mAP value in our experiments (32.4%). The decreased number of documents seems to increase the difficulty of the retrieval. The mAP for GRK-110 is slightly better than GRK-120 (39.5% vs 39.4%), indicating that we remove pages that are hard to retrieve or contain less handwriting - with an increase of 6% with respect to Top-1 accuracy. The difference in performance for both subsets when training on the full dataset is less than 1% mAP.

Table 4. Baseline results on GRK-69, GRK-110 and GRK-120.

Test set	Train set	Top-1	Top-5	Top-10	mAP
GRK-69	GRK-69	50.0	59.4	71.0	32.4
	GRK-120	48.3	60.9	71.0	32.3
GRK-110	GRK-110	62.5	79.1	81.0	39.5
	GRK-120	61.5	78.2	84.5	40.0
GRK-120	GRK-120	56.5	74.2	80.0	39.4

5.2 Character-based Aggregation

The method proposed in this paper is character-based aggregation instead of using SIFT patches. We use the same networks for aggregating the features of the respective samples.

*Kai-**s**.* Firstly, we evaluate the annotated *kai*-s, provided with a quality label (`BTx`), and present the retrieval results in Table 5. We observe a performance gain of +2.4% with respect to the main metric, mAP, on the GRK-110 dataset. However, the Top-x accuracies still trail. Therefore, we also check the precision of the first five/ten documents, where the *kai*-s outperform the SIFT approach, which shows that we have more documents retrieved on top of the list. Secondly, considering the amount of data used to calculate the global page descriptor for each document - about 2.6k per page for random SIFT patches vs. 11 *kai*-s - our experiment shows that the actual text influences the performance. By using pre-defined characters as input, in our case *kai*-s, we improve the retrieval while also reducing the amount of data needed. Finally, we find that the retrieval is able to benefit even from *kai*-s of lower quality (e.g. damaged ones), where we achieve the highest mAP for considering all samples (BT{1,2,3}). We argue that this might be due to pages containing a small amount of samples, and adding data might be beneficial, even if parts of the character are not fully available.

Table 5. Retrieval results on GRK-110 for using *kai*-s or SIFT patches (Baseline).

Kais	Top-1	Top-5	Top-10	Pr@5	Pr@10	mAP	Samples per page
BT1	47.0	67.3	75.0	31.7	24.8	38.6	6.5
BT{1,2}	49.0	70.0	79.0	**36.0**	29.4	41.1	11.0
BT{1,2,3}	50.0	67.3	76.4	35.8	**29.8**	**41.9**	11.5
Baseline	**62.5**	**79.1**	**81.0**	35.8	26.2	39.5	2614.8

*Baselines for Kai**-s**.* Secondly, we evaluate our character-based approach on other baseline methods as well, namely different encodings and aggregations. Since there is no independent training set available, we rely on ResNet56 trained via Cl-S [1] as an unsupervised feature extractor. The results for NetVLAD, SumPooling, Generalized Max Pooling (GMP) [11] and VLAD are shown in Table 6. They show that the aggregation of *kai*-s performs on par or slightly outperforms the SIFT patches in terms of mAP, with VLAD being the only method where the SIFT patches work better. We assume that this is mainly because the training set used for the vocabulary only consists of the random SIFT patches, with no training set for the *kai*-s available. This also demonstrates the benefit of the integrated codebook learned during training of NetVLAD, which seems to generalize better than the codebook of VLAD, generated by k-means. However, for Greek papyri, advanced encoding strategies such as NetVLAD or VLAD are not significantly better than just SumPooling. Furthermore, the results show that *kai*-s are able to eliminate the need for data, with only 11 *kai*-s performing similarly to 2.6k SIFT patches per page.

Kai-s and Other Characters. Additionally to the study of the *kai*-s, we provide preliminary results for combining the *kai*-s with the letters mentioned in the methodology: $\varepsilon, \kappa, \mu, \omega$. The dataset used is the GRK-69 subset. We experiment with aggregating all possible permutations of the five characters and report single character performances as well as the best/worst five combinations of the characters in Table 7. Firstly, as shown in Table 7a, the use of the *kai*-s also outperforms the baseline of using SIFT patches by 3.2%. Other characters perform worse, which we assign to the lack of data or the fact that they are easily distorted by the previous or following character (due to the cursivity of the handwriting). However, evaluating combinations of different letters, we find that we are able

Table 6. Other baseline results on GRK-110 for using *kai*-s or SIFT patches.

	Baseline		Kais	
	Top-1	mAP	Top-1	mAP
NetVLAD	**62.5**	39.5	50.0	41.9
SumPooling	56.7	**42.9**	**61.5**	**43.0**
GMP	56.7	42.8	**61.5**	42.8
VLAD	56.8	42.1	39.4	31.0

to boost the retrieval performance, with the combination of *kai*-s and κ performing the best (36.2% mAP, Table 7b). For a fair overview, we also give the five worst combinations, for which none of them includes the *kai*-s. We conclude that the *kai*-s contain the most discriminating power for WR, but annotating or automatically detecting specific characters for aggregation improves the performance, even if we have less data available, e.g., our dataset only includes 287 samples of μ, averaging only four samples per document.

Table 7. Retrieval results on GRK-69 for using different characters or SIFT patches (Baseline). (a) Performance when only using one type of character. (b) Combinations with best performance. (c) Combinations with worst performance.

(a)

Character	mAP
Kai	**35.6**
ε	23.6
μ	24.9
κ	25.1
ω	22.3
Baseline	32.4

(b)

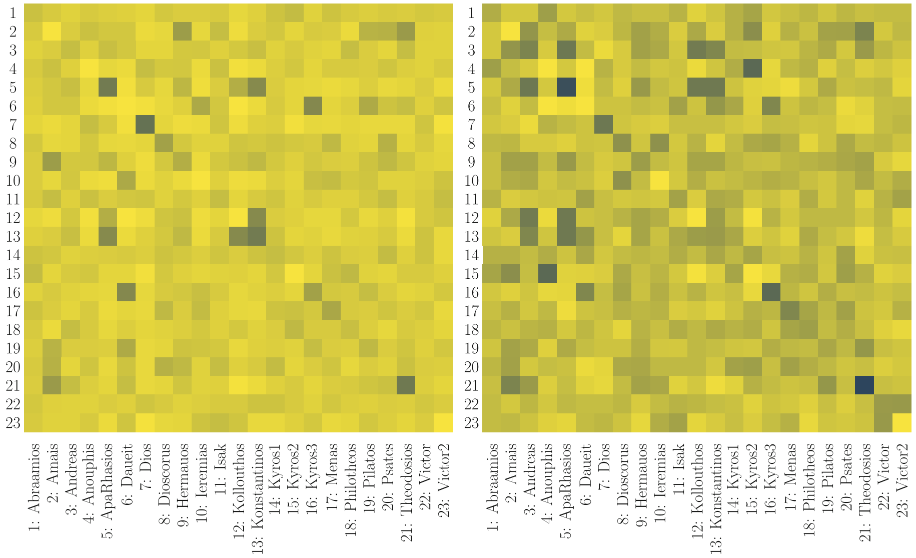
Characters	mAP
Kai, κ	**36.2**
Kai, κ, μ	35.9
Kai, μ	35.2
Kai, ω	34.1
Kai, ε, κ, μ	32.9
Baseline	32.4

(c)

Characters	mAP
ε, κ, μ	24.0
ε, κ, ω	25.1
ε, μ	25.3
ε, ω	25.7
κ, ω	26.3
Baseline	**32.4**

5.3 Qualitative Analysis

We conclude the evaluation of our paper by providing two qualitative studies, both concerning the performance of the *kai*-s. Firstly, we qualitatively compare the writer similarity for SIFT patches and the *kai*-s. We average the pairwise similarity of all documents (excluding pairs for the same documents), and provide a heatmap in Fig. 5. By comparing the matrices, we observe that some writers have high similarities in both, e.g. Theodosius, where in particular the *kai*-s are highlighted, or Apa Rhasios. However, the intra-class similarity of the *kai*-s is higher for writers like Andreas or Victor than aggregating SIFT patches. This



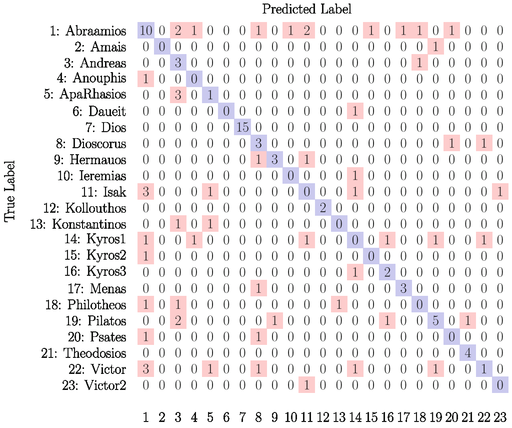
(a) SIFT patches

(b) Kais

Fig. 5. Similarity matrices per writer (pairwise document similarity averaged) when using (a) SIFT patches and (b) only *kai*-s for calculating the global page descriptor. The darker the square, the higher the similarity.

observation is of interest for specialists of ancient handwritings (paleographers) because it illustrates the potential of our approach to contribute characterizing intra- and inter- writer variations in human understandable terms. In general, we also observe multiple instances of similarities across writers in both matrices, which invites papyrologists to examine if indeed those writers have similar styles. It might also be due to a lack of data for the *kai*-s and the general performance in terms of mAP, that is only about 40%. We find that the aggregation of *kai*-s leads to additional similarities in the feature space for some writers rather than just higher similarities compared to the SIFT patches, indicating that an extension of our approach might consider both aggregation schemes for better performance.

Secondly, we provide a confusion matrix of the WR performance of the *kai*-s where the "predicted label" is the writer of the most similar writing sample, as shown in Fig. 6. With this, we can identify high similarities between pairs of writing samples. In GRK-120, 10 writers have only one document, but some are long enough to be split into several images or writing samples (from 2 to 15). Interestingly, the writers who have the highest scores, e.g. all their writing samples correctly attributed, are writers attested in one single document of large size (Dios and Theodosios with respectively 15 and 4 writing samples). The five writers that are attested in one single but short papyrus, Amais, Anouphis, Daueit, Kyros2, Victor2, are not correctly recognized (see also Fig. 2 for the



Predicted Label

True Label

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1: Abraamios	10	0	2	1	0	0	0	1	0	1	2	0	0	0	1	0	1	1	0	1	0	0	0
2: Amais	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
3: Andreas	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
4: Anouphis	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5: ApaRhasios	0	0	3	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6: Daueit	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
7: Dios	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8: Dioscorus	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
9: Hermauos	0	0	0	0	0	0	0	1	3	0	1	0	0	0	0	0	0	0	0	0	0	0	0
10: Ieremias	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
11: Isak	3	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
12: Kollouthos	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
13: Konstantinos	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14: Kyros1	1	0	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0	0	1	0
15: Kyros2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16: Kyros3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0
17: Menas	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0
18: Philotheos	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
19: Pilatos	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	5	0	1	0	0
20: Psates	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21: Theodosios	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0
22: Victor	3	0	0	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	1	0
23: Victor2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 6. Confusion matrix of the writers for each document considering Top-1 accuracy when aggregating the *kai* features.

amount of annotations per writer). At the opposite, some writers attested on many documents over a long period of time, like Abraamios and Victor1, are not well recognized either. Thus the method seems to work very well when there is enough data to grasp the original features of one document but does not manage to recognize correctly a hand that probably changes over time (and possibly writing implements). We also note that while some confusions can be explained by chronology such as Apa Rhasios being incorrectly classified three times as Andreas, a coeval notary, some others are with much older or later writers. For instance Dioscorus, who is supposed to have an easy-to-distinguish handwriting, is correctly recognized 3 times but confused once with Victor1 and another time with Psates, who are much earlier. It could be a way for paleographers to spot writers using "archaic" or "innovative" styles.

6 Conclusion

This study tackles the challenges of identifying writers in Greek papyri, for which we introduced a new method that focuses on individual characters and found it improves the WR performance while reducing also the need of data. We find that the *kai*-s are the most discriminative resource used for feature aggregation, but the WR can further be boosted by using other letters (κ or μ). Our dataset is publicly available, and we also provided qualitative studies on our approach compared to SIFT patches, the current methodology widely used for WR. One drawback of our method, the need for a scholar to annotate characters necessary for aggregation, could be overcome by further investigating character localization and classification, which is already an active field of research [21]. We think combining our pipeline with a detector is the next step to aggregate all characters of a document for aggregation. It is also worth investigating the aggregation of other n-grams, not just *kai*-s. Additionally, this paper only considered simple aggregation techniques, but for future work, our paper opens the field of advanced schemes for aggregating the features of the characters, for example, using graph- or cluster-based methods. In the field of papyrology, WR, in particular our character-based approach, is a promising tool to find similarities between documents that have lost information about their context of production (writers but also schools of writers, places and periods), thus allowing major improvements in our understanding of Greco-Roman Egypt.

References

1. Christlein, V., Gropp, M., Fiel, S., Maier, A.K.: Unsupervised feature learning for writer identification and writer retrieval. In: 14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, 9-15 November 2017, pp. 991–997 (2017)
2. Christlein, V., Maier, A.K.: Encoding CNN activations for writer recognition. In: 13th IAPR International Workshop on Document Analysis Systems, DAS 2018, Vienna, Austria, 24–27 April 2018, pp. 169–174 (2018)
3. Christlein, V., Marthot-Santaniello, I., Mayr, M., Nicolaou, A., Seuret, M.: Writer retrieval and writer identification in Greek papyri. In: Intertwining Graphonomics with Human Movements - 20th International Conference of the International Graphonomics Society, IGS 2021, Las Palmas de Gran Canaria, Spain, 7–9 June 2022, Proceedings, vol. 13424, pp. 76–89 (2022)
4. Cilia, N.D., et al.: A novel writer identification approach for Greek papyri images. In: Image Analysis and Processing - ICIAP 2023 Workshops - Udine, Italy, 11–15 September 2023, Proceedings, Part II, pp. 422–436 (2023)
5. Cilia, N.D., Stefano, C.D., Fontanella, F., Marthot-Santaniello, I., di Freca, A.S.: Papyrow: a dataset of row images from ancient Greek papyri for writers identification. In: Pattern Recognition. ICPR International Workshops and Challenges - Virtual Event, 10–15 January 2021, Proceedings, Part VII, vol. 12667, pp. 223–234 (2020)
6. Marthot-Santaniello, I., Vu, M., Serbaeva, O., Beurton-Aimar, M.: Stylistic similarities in Greek papyri based on letter shapes: a deep learning approach. In: Document Analysis and Recognition - ICDAR 2023 Workshops - San José, CA, USA, 24–26 August 2023, Proceedings, Part I, vol. 14193, pp. 307–323 (2023)
7. McArthur, H.K.: Kai frequency in Greek letters. New Testament Stud. **15**(3), 339–349 (1969)
8. Mohammed, H.A., Märgner, V., Konidaris, T., Stiehl, H.S.: Normalised local naïve bayes nearest-neighbour classifier for offline writer identification. In: 14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, 9–15 November 2017, pp. 1013–1018 (2017)
9. Mohammed, H.A., Marthot-Santaniello, I., Märgner, V.: Grk-papyri: a dataset of Greek handwriting on papyri for the task of writer identification. In: 2019 International Conference on Document Analysis and Recognition, ICDAR 2019, Sydney, Australia, 20–25 September 2019, pp. 726–731 (2019)
10. Morton, A.Q.: The authorship of Greek prose. J. Royal Stat. Soc. Series A (General) **128**(2), 169–233 (1965)
11. Murray, N., Perronnin, F.: Generalized max pooling. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, 23–28 June 2014, pp. 2473–2480 (2014)

12. Nasir, S., Siddiqi, I., Moetesum, M.: Writer characterization from handwriting on papyri using multi-step feature learning. In: Document Analysis and Recognition, ICDAR 2021 Workshops, Lausanne, Switzerland, 5–10 September 2021, Proceedings, Part I, vol. 12916, pp. 451–465 (2021)
13. Pavlopoulos, J., et al.: Explaining the chronological attribution of Greek papyri images. In: Discovery Science - 26th International Conference, DS 2023, Porto, Portugal, 9–11 October 2023, Proceedings (2023)
14. Peer, M., Kleber, F., Sablatnig, R.: Writer retrieval using compact convolutional transformers and netmvlad. In: 26th International Conference on Pattern Recognition, ICPR 2022, Montreal, QC, Canada, 21–25 August 2022, pp. 1571–1578 (2022)
15. Peer, M., Kleber, F., Sablatnig, R.: Towards writer retrieval for historical datasets. In: Document Analysis and Recognition - ICDAR 2023 - 17th International Conference, San José, CA, USA, 21–26 August 2023, Proceedings, Part I, pp. 411–427 (2023)
16. Peer, M., Sablatnig, R.: Feature mixing for writer retrieval and identification on papyri fragments. In: Proceedings of the 7th International Workshop on Historical Document Imaging and Processing (2023)
17. Pirrone, A., Beurton-Aimar, M., Journet, N.: Self-supervised deep metric learning for ancient papyrus fragments retrieval. Int. J. Document Anal. Recognit. **24**(3), 219–234 (2021)
18. Pratikakis, I., Zagoris, K., Karagiannis, X., Tsochatzidis, L., Mondal, T., Marthot-Santaniello, I.: Icdar 2019 competition on document image binarization (dibco 2019). In: 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 1547–1556 (2019)
19. Rasoulzadeh, S., BabaAli, B.: Writer identification and writer retrieval based on netvlad with re-ranking. IET Biom. **11**(1), 10–22 (2022)
20. Serbaeva, O., White, S.: Read for solving manuscript riddles: a preliminary study of the manuscripts of the 3rd ṣaṭka of the Jayadrathayāmala. In: Document Analysis and Recognition – ICDAR 2021 Workshops, Lausanne, Switzerland, 5–10 September 2021 Proceedings, Part 2, pp. 339–348 (2021)
21. Seuret, M., et al.: ICDAR 2023 competition on detection and recognition of Greek Letters on Papyri, pp. 498–507 (2023)

Beyond Relevant Documents: A Knowledge-Intensive Approach for Query-Focused Summarization Using Large Language Models

Weijia Zhang[1](✉), Jia-Hong Huang[1], Svitlana Vakulenko[2], Yumo Xu[3], Thilina Rajapakse[1], and Evangelos Kanoulas[1]

[1] University of Amsterdam, Amsterdam, Netherlands
w.zhang2@uva.nl
[2] Amazon, Barcelona, Spain
[3] University of Edinburgh, Edinburgh, Scotland

Abstract. Query-focused summarization (QFS) is a fundamental task in natural language processing with broad applications, including search engines and report generation. However, traditional approaches assume the availability of relevant documents, which may not always hold in practical scenarios, especially in highly specialized topics. To address this limitation, we propose a novel knowledge-intensive approach that reframes QFS as a knowledge-intensive task setup. This approach comprises two main components: a retrieval module and a summarization controller. The retrieval module efficiently retrieves potentially relevant documents from a large-scale knowledge corpus based on the given textual query, eliminating the dependence on pre-existing document sets. The summarization controller seamlessly integrates a powerful large language model (LLM)-based summarizer with a carefully tailored prompt, ensuring the generated summary is comprehensive and relevant to the query. To assess the effectiveness of our approach, we create a new dataset, along with human-annotated relevance labels, to facilitate comprehensive evaluation covering both retrieval and summarization performance. Extensive experiments demonstrate the superior performance of our approach, particularly its ability to generate accurate summaries without relying on the availability of relevant documents initially. This underscores our method's versatility and practical applicability across diverse query scenarios.

Keywords: Query-focused summarization · Knowledge-intensive tasks · Large language models

1 Introduction

Query-focused summarization (QFS) is a pivotal task with wide-ranging applications, spanning fields such as search engines and report generation [25]. It

W. Zhang, J.-H. Huang—Equal contribution.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 89–104, 2025.
https://doi.org/10.1007/978-3-031-78495-8_6

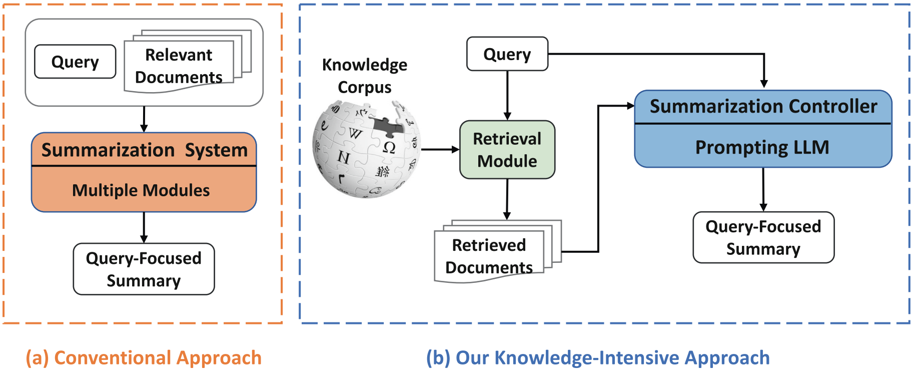


Fig. 1. The Comparison between (a) the conventional approach and (b) our knowledge-intensive approach. The conventional one assumes relevant documents are available. We aim to retrieve such documents from a large knowledge corpus.

involves analyzing a textual query alongside a collection of relevant documents to automatically produce a textual summary closely aligned with the query [5,24,34,35]. This process aims to offer users relevant insights in a condensed format by applying information compression techniques to the provided document set [17,20,35]. The conventional approach to this task assumes the availability of a set of relevant documents and creates the summary based on the information within this document set. However, in practical scenarios, this assumption may not always hold.

Consider highly specialized or niche topics as an illustration. In fields such as advanced scientific research, specialized technology, or obscure hobbies, documentation may be limited or fragmented, particularly for cutting-edge or esoteric subjects. Similarly, think about future predictions or speculations. Queries related to future events, trends, or predictions may lack existing documents as they entail speculation or anticipation of events yet to unfold. For instance, questions about the outcome of upcoming elections, the impact of emerging technologies, or the trajectory of financial markets might not be documented until after these events have occurred. In the aforementioned practical scenarios, the traditional setup of QFS with compression techniques may not work well due to the scarcity or absence of relevant documents.

To tackle this challenge, our approach reframes QFS as a knowledge-intensive (KI) task setup [28]. Unlike the traditional setup where relevant documents are assumed to be readily available, our knowledge-intensive approach only assumes access to a large-scale knowledge corpus. Our proposed approach comprises a retrieval module and a summarization controller. Given a textual query as the initial input, the retrieval module returns potentially relevant documents from the knowledge corpus, eliminating the dependence on pre-existing document sets. These retrieved documents, along with the query, are then forwarded to the summarization controller. The summarization controller harmoniously merges an advanced large language model (LLM)-powered summarizer with a carefully designed prompt, guaranteeing the produced summary is both comprehensive and directly addresses the query. To illustrate the aforementioned process, we make a comparison between the conventional approach and our proposed approach in Fig. 1.

To validate the effectiveness of our proposed approach within the KI setup, we have introduced a specialized dataset by combining existing QFS datasets [5] with three various knowledge corpora. We also enhance the dataset with human-annotated relevance labels to facilitate comprehensive evaluation for both retrieval and summarization performance. This dataset serves as the foundation for a fair evaluation of our approach and baseline models. Extensive experiments conducted in this work confirm the effectiveness of our proposed method, surpassing baseline models and showcasing its superiority in generating query-focused summaries. Notably, our approach stands out for its ability to generate accurate summaries without relying on the availability of relevant documents at the initial input stage. This underscores the versatility of our method in addressing diverse query scenarios, further validating its practical applicability.
The primary contributions of this work can be summarized as follows:

1) **Introduction of a Knowledge-Intensive Approach**: We introduce a new knowledge-intensive approach for QFS, departing from the conventional paradigm that relies on the availability of relevant documents in the initial input stage. This innovative method involves the direct retrieval of pertinent documents from an extensive knowledge corpus, leveraging the provided textual query as the basis.
2) **Development of a Summarization Controller**: We propose a summarization controller that orchestrates the process of generating query-focused textual summaries using an integrated LLM-based summarizer with a specially designed prompt. This controller streamlines the summarization process, ensuring alignment with the query intent.
3) **Creation of a Specialized Dataset**: To assess the effectiveness of the proposed approach, we introduce a new dataset explicitly designed for knowledge-intensive QFS. This dataset, along with human-annotated relevance labels, serves as a standardized benchmark for evaluating both retrieval and summarization performance.
4) **Demonstration of Superior Performance**: We present extensive experimental results showcasing the superior performance of the proposed approach compared to baseline models. Particularly noteworthy is the approach's capability to generate accurate summaries without the prerequisite of available relevant documents at the initial input stage. This feature underscores the method's versatility and practical utility across a range of query scenarios.

2 Related Work

In this section, we cover three lines of research: query-focused summarization, knowledge-intensive language tasks, and open-domain summarization, all related to our study.

Query-Focused Summarization. Query-focused summarization aims to generate a summary tailored to a given query from a set of topic-related contexts [5,24,40]. Conventional QFS methods primarily focused on sentence-level extraction techniques to eliminate redundant information [17,34,35]. However, they heavily rely on pre-existing relevant documents and are typically developed for a limited set of closed-domain documents (∼50 documents). In contrast to all previous work, our work does not assume the existence of relevant documents to the given query. Instead, our approach handles open-domain document-level retrieval, aiming to identify a small set of relevant documents from a pool of millions of candidates, accommodating more diverse topics and contexts. Another line of research aims to create larger QFS datasets to alleviate data scarcity automatically, utilizing information retrieval or clustering methods [15,24]. However, these datasets lack a standardized benchmark to assess the retrieval performance, as they lack relevance annotations for retrieved documents. In contrast, our dataset includes human annotations for the relevance of retrieved documents. This facilitates retrieval evaluation and provides insights into understanding the effects of retrieval errors on summarization performance.

Knowledge-Intensive Tasks. Knowledge-intensive language tasks [3,28] involve addressing user needs by leveraging a large-scale knowledge corpus. One of the most related KI tasks is the long-form question answering task (LFQA) [7,32], where the goal is to provide comprehensive answers to given questions. Compared to QFS, the key difference between LFQA and QFS lies in their objectives and the role of input documents. LFQA provides detailed answers to complex questions, whereas documents are optional external sources to enhance answer quality [38,39]. In contrast, QFS requires a concise summary from multiple documents, tailored to a specific query. In this case, the relevant documents are essential inputs to generate the summary. Furthermore, existing LFQA tasks either struggle with accurately grounding their answers in the provided documents [14] or they heavily rely on the inherent ambiguity present in questions [26]. In comparison, our work has two significant advantages: 1) Our approach ensures generated query-focused summaries are firmly based on the content of the retrieved documents; and 2) The queries are unambiguous and accurately reflect complex information needs in QFS, resulting in naturally longer summaries to address various information needs adequately.

Open-Domain Summarization. Preliminary studies [41,42] mark the initial phase of investigating the complexities of open-domain QFS. Building upon these, recent studies [10,45] have embarked on similar investigations. They either leverage multi-document summarization (MDS) datasets [6] or adapt query-based single-document summarization datasets [33,44]. In comparison, this work has two significant differences: 1) Previous studies do not include human annotations to judge the relevance of retrieved documents, making their evaluations less reliable. We add human annotations to improve the accuracy of these evaluations. 2) Previous studies mostly analyze documents from specific areas like news or medicine, limiting their scope. Our study expands this by including both internal documents and a larger external knowledge base like Wikipedia, making our findings more applicable to real-world situations.

3 Knowledge-Intensive Approach

In this section, we formulate our knowledge-intensive task setup and detail our proposed approach.

3.1 Task Formulation

To better understand the difference between traditional QFS and our knowledge-intensive setup, we first describe the traditional QFS and then formulate our proposed knowledge-intensive setup.

Traditional QFS. Let the tuple $(q, \mathcal{D}, s)$ denotes an example in the traditional QFS task, where q is a given query, $\mathcal{D} = \{d_1, \ldots, d_m\}$ is a set of relevant documents and s is the gold summary for the document set $\mathcal{D}$ and the query q. The goal of QFS is to produce the query-focused summary given the documents and the query as input: $(q, \mathcal{D}) \rightarrow s$. It is worth noting that $\mathcal{D}$ is collected manually by human annotators and usually includes tens of documents.

Knowledge-Intensive QFS. In this work, we reframe the QFS task to a knowledge-intensive task setup as $q \rightarrow s$. As shown in Fig. 1, we do not rely on the relevant document set $\mathcal{D}$. Instead, we assume the access to a large-scale knowledge corpus: $\mathcal{K} = \{d_1, \ldots d_n\}$. In practice, $\mathcal{K}$ consists of millions of documents, i.e., $n \gg m$. It is worth noting that our goal remains to generate the query-focused summary s with respect to the query q. However, different from the traditional setup, there is no guarantee that the provided documents are relevant to the query. Instead, there are millions of irrelevant documents in the knowledge corpus. Thus, an effective information retrieval model is necessary to supplement the extended setup, which is described below.

3.2 Methodology

Our approach includes two components: a retrieval module and a summarization controller, as illustrated in Fig. 1. The retrieval module returns top-k documents from the knowledge corpus based on the given query. Then the summarization controller takes the query and retrieved documents as the inputs to generate a query-focused summary.

Prompt 3.1: Summarization Controller

Instruction: You will be given a query and a set of documents. Your task is to generate an informative, fluent, and accurate query-focused summary. To do so, you should obtain a query-focused summary step by step.

Step 1: Query-Relevant Information Identification
In this step, you will be given a query and a set of documents. Your task is to find and identify query-relevant information from each document. This relevant information can be at any level, such as phrases, sentences, or paragraphs.

Step 2: Controllable Summarization
In this step, you should take the query and query-relevant information obtained from Step 1 as the inputs. Your task is to summarize this information. The summary should be concise, include only non-redundant, query-relevant evidence, and be approximately 250 words long.

Demonstrations:
Few-shot human-written demonstrations.

Query: {*Input query*}
Documents: {*Retrieved documents*}

Retrieval Module. Given a query q and a document d_i from the knowledge corpus $\mathcal{K}$, the retrieval module first estimates the relevance $Rel(q, d_i)$ between q and d_i and then ranks all documents based on their relevance scores. Typically, we only consider top-k retrieved documents. There are two main categories of retrieval models: sparse and dense models [43]. The former estimates relevance between the query and the document using weighted counts of their overlapping terms. The latter first transforms both the query and the document into vectors within an embedding space. The relevance is then computed using the dot product of their respective vectors. In this work, we explore representative sparse and dense models, which are BM25 [30] and Dense Passage Retrieval (DPR) [13], respectively.

Summarization Controller. After we obtain top-k retrieved documents from the retrieval module, we feed the query and the documents into the summarization controller to generate a query-focused summary. As large language models (LLMs), such as GPT-3.5 [27], have exhibited remarkable generation ability in many text generation tasks [16], we introduce a summarization controller by prompting a LLM to generate the summary. As shown in Prompt 3.1, the controller involves two primary steps: identifying query-relevant information and generating a controllable summary. First, to ensure the summary's relevance to the query, we explicitly instruct the LLM to identify query-relevant information within the documents. It is worth noting that the LLM can determine information at various levels, including phrases, sentences, and paragraphs. Subsequently, we instruct the LLM to write a controllable summary that meets the length limit of 250 words based on the query-relevant information. This ensures the summary is sufficiently comprehensive and relevant to the query. Moreover, we include few-shot demonstrations in the prompt to enable in-context learning for the LLM, which has been proven to be effective in many natural language processing tasks [2].

4 KI-QFS Dataset

In this section, we describe dataset collection and relevance annotation process for our knowledge-intensive setup. The dataset includes a collection of query-summary pairs and three alternatives of knowledge corpora. The relevance annotation aims to facilitate retrieval evaluation.

4.1 Dataset Collection

Collecting Query-Summary Pairs. We build our dataset on the top of query-summary pairs (q, s) on existing QFS resources. Specifically, we adopt DUC 2005–07 [5], three standard QFS benchmark datasets.[1] The DUC datasets consist of three subsets collected for the Document Understanding Conferences (DUC) from 2005 to 2007. Each subset contains 45–50 clusters. Each cluster (or a topic) contains a query, a set of topic-related documents, and multiple reference summaries. As relevant documents are inaccessible in our dataset, we only collect query-summary pairs first. The relevant documents will be used to build a knowledge corpus, which is described below. For data split, we follow previous work [17,21] to use the pairs of the first two years (2005–2006) as the training set and randomly select 10% from the training set as the validation set. We leave the third subset (2007) as the test set. Finally, the training set, validation set, and test set contain 90, 10, and 45 examples, respectively.

Building Knowledge Corpora. We explore three alternatives of knowledge corpora $\textsc{Corpus}$ for the query-summary pairs above. Specifically, we first follow the standard data processing for large-scale knowledge sources [13] to collect documents, where we split each origin document into non-overlapping context documents with 100 words maximum.[2] The knowledge corpora are as follows: 1) The first knowledge corpus, $\textsc{Corpus}_{Int}$, is an internal collection, where we take all clusters of DUC documents from the three years to form this collection, which results in about 32K documents in total. As the queries have an

[1] We take DUC as an example, but nothing prohibits exploring other QFS resources.

[2] Some studies [13] also use passages to name the processed text chunks, but we adhere to the term "documents" to maintain the coherence of the paper.

explicit connection with the documents, this collection can be considered an in-domain knowledge corpus, where relevant documents are relatively easy to find. 2) However, as our main goal is to explore knowledge-intensive QFS on the large-scale knowledge corpus, we consider the second external knowledge corpus named Corpus$_{Ext}$. We use the Wikipedia dump in the KILT benchmark [28] to form this corpus, resulting in about 21 million documents in total. 3) However, as Corpus$_{Ext}$ is a Wiki-based corpus, there is no guarantee that the collection contains sufficient evidence to answer the query since the reference summary is derived from the content of the original DUC documents. To this end, we build an augmented knowledge corpus called Corpus$_{Aug}$ by combining the previous two collections. We merge all the documents of Corpus$_{Int}$ and Corpus$_{Ext}$ into this collection ensuring that summaries can be grounded in the collection and that the collection is large-scale to make the task challenging.

4.2 Relevance Annotation

As the retrieval process is necessary for our setup, it is important to evaluate the performance of retrieval models so we can better analyze their effect on summarization. However, the DUC datasets do not come with relevance labels to designate which document provides evidence for the final summary. A possible solution to that would be to do a lexical match between the individual summary sentences and the available document collection, however, such an automatic evaluation has its disadvantages (e.g. a summary sentence may appear in a document but outside the right context, lexical overlap may pay attention to insignificant words in the summary, and there may be a lexical gap between semantically similar sentences). Therefore, we collect human annotations through Amazon Mechanical Turk (MTurk) to create our evaluation set.

As the main goal of the annotation process is to obtain relevant documents that support the summary, we begin with filtering out irrelevant documents, which are the majority of millions of candidate documents in the knowledge corpus. Following the TREC document retrieval task [4], we apply retrieval models to choose top-ranked documents, as discussed in Subsect. 3.2. We first perform both BM25 and DPR on Corpus$_{Int}$ and Corpus$_{Ext}$, and construct top-50 pools, which result in a maximum of 200 candidate documents for each query. As there are 45 queries in our test set, we end up with 7761 query-document pairs to be annotated. We then design a web annotation interface shown to MTurk workers, which contains a query, a reference summary, and at most five documents. We ask workers to label whether a document contains either part of the summary or evidence to a query, adapting the annotation instructions in [4]. The judgments are on a 4-point scale from 0 to 3: (i) **0-irrelevant**: the document has nothing to do with the query; (ii) **1-weakly related**: the document seems related to the query but fails to contain any evidence in the summary; (iii) **2-related**: the document provides unclear information related to the query, but human inference may be needed; and (iv) **3-relevant**: the document explicitly contains evidence that is part of the summary. Prior to the collection of the annotations we performed two pilot studies to improve our annotation interface and the instructions. We collect three annotations per document and use a majority vote to determine the final relevance label. The distribution of relevance labels is 837 relevant, 5811 related, 1097 weakly related, and 16 irrelevant documents. We also measure inter-annotator agreement (IAA) using Fleiss Kappa [8]. A Fleiss Kappa score of 0.42 was obtained, which indicates a moderate agreement.

5 Evaluation Metrics

In this section, we describe evaluation metrics. As our approach includes a retrieval module and a summarization controller, we evaluate both components using suitable evaluation metrics.

5.1 Retrieval Evaluation

To evaluate retrieval effectiveness, i.e., the ability of the retrieval module to identify and retrieve evidence to be used to compose the summary, we use standard information retrieval (IR) metrics, and in particular precision@k (P@k) and recall@k (R@k) where k denotes the cut-off. For P@k and R@k, we map the human-annotated judgment level 3 to positive and judgment levels 0–2 to negative and report the corresponding results for $k \in \{10, 50\}$.

5.2 Summarization Evaluation

Lexical Metrics. In our experiments, we measure the accuracy of generated summaries against reference summaries. Following previous work [17,35], we employ the commonly-used standard ROUGE-1, ROUGE-2, and ROUGE-SU4 [22], which evaluate the accuracy on uni-grams, on bi-grams, and on bi-grams with a maximum skip distance of 4, respectively. We report the F_1 score with a maximum length of 250 words for the metrics. It is worth noting that there are multiple reference summaries on our dataset, we take the average of the ROUGE scores for all summaries as the final reported ROUGE score.

Semantic Metrics. However, ROUGE-based metrics only measure lexical overlap between generated summaries and reference summaries, failing to capture their semantic similarities. Recent semantic metrics, such as BERTScore [37] and BARTScore [36], have been shown to be effectively correlated with human judgments [11]. Therefore, we also report them in our experiments. Specifically, we report the F1 score of BERTscore and use recommended `deberta-xlarge-mnli` as the backbone. For BARTScore, we use the recall version of BARTScore as it shows more effective performance in summarization tasks [36], where we use `bart-large-cnn` as the backbone. It is worth noting that the raw scores of BARTScore are negative log probabilities that are difficult to explain, we follow previous work [31] to normalize them to positive scores with an exponential function.

6 Experimental Setup

In this section, we describe the baseline models for comparison in our experiments. We then discuss the implementation details.

6.1 Baselines

We compare our proposed method with the following two families of baseline models:

Weakly Supervised QFS Models. To illustrate new challenges brought by our knowledge-intensive setup, we first include two weakly supervised models designed for traditional QFS tasks: 1) QuerySum [34]: an extractive QFS model leveraging distant supervision signals from trained question answering (QA) models to select salient sentences as the summary. 2) MargeSum [35]: an abstractive QFS model that employs generative models trained on generic summarization resources to boost sentence ranking and summary generation, achieving competitive performance in the weakly supervised setup. We discuss the adaption of two models to our knowledge-intensive setup in Subsect. 6.2.

Supervised Retrieval-Augmented Models. We employ supervised Retrieval-Augmented Generation (RAG) models below: 1) RAG-Sequence [19]: a generative model in which document retrieval and summary generation are learned jointly. As RAG models have two variants including RAG-Token and RAG-Sequence, we mainly report the results of RAG-Sequence as it shows better performance in knowledge-intensive tasks. Specifically, the RAG-Sequence model employs DPR [13] for retrieval and BART$_{\text{large}}$ [18] for generation. 2) Fusion-in-Decoder (FiD) [12]: an encoder-decoder architecture which has achieved state-of-art performance in knowledge-intensive tasks [1]. In this model, encoded representations for retrieved documents are first concatenated and then fed into the decoder to generate the output. Following the origin paper, we use T5$_{\text{base}}$ [29] as the base model.

LLM-Based RAG Models. In addition to supervised RAG models, we compare our approach with LLM-based RAG models [9]. Specifically, we employ BM25 or DPR for retrieval and GPT-3.5 for generation, referring to this baseline as NaiveRAG. Following the prompting strategy from Gao et al. [9], we instruct GPT-3.5 to answer a given query based on the associated retrieved documents. Notably, NaiveRAG does not require fine-tuning.

6.2 Implementation Details

For retrieval models, we use the library Pyserini [23] to implement BM25 and DPR.[3] For the summarization controller, we employ GPT-3.5 using the Ope-

[3] https://github.com/castorini/pyserini.

Table 1. Retrieval evaluation of different models on the test set of our dataset over three knowledge corpora. P@k and R@k denote precision and recall regarding cutoff k. The best results are in **bold**. *means that the improvement is statistically significant (a two-paired t-test with p-value < 0.01).

Corpus	Model	P@10	P@50	R@10	R@50
Corpus$_{Int}$	BM25	**16.0***	**12.0**	**8.2***	**30.2***
	DPR	11.6	11.5	6.0	27.8
Corpus$_{Ext}$	BM25	12.7	8.6	**7.0**	22.2
	DPR	**13.8***	**12.6***	6.9	**29.7***
Corpus$_{Aug}$	BM25	12.2	8.9	6.8	23.2
	DPR	**14.4***	**12.5***	**7.4**	**30.3***

nAI API service, where we use long-context version `gpt-3.5-turbo-0613`.[4] We utilize 3-shot demonstrations in few-shot settings and present results for both zero-shot and few-shot settings. The temperature and top-p are set to 0.1 and 0.95, respectively, with a maximum output length of 400 tokens. The number of top-k retrieved documents is fixed at 50. Notably, we employed the long-context version of GPT-3.5, which supports a maximum context window of 16,385 tokens. Given that each retrieved document contains up to 100 words, when we concatenate 50 documents into a prompt, the total length of the prompt is roughly $7,118$ tokens on average, which is comfortably below the maximum limit.

For QFS models, as they have achieved competitive performance on the QFS datasets in a weak supervision manner, we do not fine-tune them on our dataset. For inference, although they include a retrieval module that performs in-domain evidence estimation, their retrieval models are not designed to handle large-scale, open-domain knowledge bases, such as Corpus$_{Ext}$. Therefore, we first retrieve top-1000 documents using BM25 for each query and then feed them as inputs to their customized sentence extraction modules. We follow the original papers [34,35] to set their hyper-parameters.

For retrieval-augmented models, as they are originally designed for short-form open-domain QA [3]. For fair comparisons, we fine-tune them on our training set. We follow the original papers [12,19] to set their associated hyper-parameters and training setups. For inference, we set the maximum decoded length and beam size to 250 and 5, respectively. All experiments were conducted on a single A6000 GPU.

7 Results and Analyses

Our experimental results primarily address the following research questions (RQs):

RQ1 What is the retrieval effectiveness of different retrieval models in the knowledge-intensive setup?

[4] https://platform.openai.com.

RQ2 How does our approach compare to baseline models in the knowledge-intensive setup?

RQ3 What is the effect of our knowledge-intensive setup compared to the traditional QFS?

7.1 Retrieval Results

To answer **RQ1**, we evaluate the retrieval effectiveness of BM25 and DPR on the test set of our dataset among three different knowledge corpora. The results are shown in Table 1. We find that BM25 outperforms DPR on Corpus_{Int}. One plausible explanation is that the internal knowledge corpus (Corpus_{Int}) comprises documents sourced from human-curated DUC datasets, potentially containing more relevant keywords. These keywords can be easily retrieved by BM25 which is a term-based IR method. Conversely, DPR exhibits better performance on Corpus_{Ext}. This is because it is pre-trained on Wikipedia data dumps, allowing it to better capture semantic representations of Wikipedia documents, thus enhancing retrieval performance. Moreover, we observe DPR's superior performance over BM25 on Corpus_{Aug}. This could be due to the high similarity between Corpus_{Aug} and Corpus_{Ext}, as Corpus_{Aug} only contains a small fraction of documents from DUC datasets and both predominantly consist of Wikipedia documents. However, the fairly low precision and recall scores across all three collections underscore challenges in such large-scale knowledge retrieval. Addressing these challenges requires more research efforts to enhance retrieval performance.

7.2 Summarization Results

To answer **RQ2**, we make a comparison between our approach and baseline models, including weakly-supervised QFS, supervised RAG models, and LLM-based RAG models. The results are shown in Table 2. Interestingly, we find our few-shot prompted approach variant surpasses all baseline models across all evaluation metrics, especially in terms of semantic metrics. For instance, our best approach variant achieved a BERTScore of 29.2, surpassing the best-performing NaiveRAG, which had a BERTScore of 25.1. This result underscores the superiority of our proposed approach. Additionally, our approach consistently outperforms baselines across three different knowledge corpora and demonstrates robustness across various retrieval models. Furthermore, when comparing zero-shot and few-shot settings, we find that the performance of our approach in the few-shot setting significantly exceeds that in the zero-shot setting. This indicates that few-shot demonstrations have a highly positive effect on model performance. Lastly, our results reveal that fine-tuned FiD models outperform QFS models, indicating the advantages of fine-tuning on our training data. However, we also observe that the performance of the supervised RAG-Sequence model was comparatively lower, potentially due to the limited size of the training data.

Table 2. Summarization evaluation of our approach and baseline models over the three knowledge corpora. R1, R2, RS, BE, and BA stand for ROUGE-1, ROUGE-2, ROUGE-SU4, BERTScore, and BARTScore, respectively. The best results are in **bold**. *indicates that the improvement to the best baseline model is statistically significant (a two-paired t-test with p-value < 0.01).

	CORPUS$_{Int}$					CORPUS$_{Ext}$					CORPUS$_{Aug}$				
Model	R1	R2	RS	BE	BA	R1	R2	RS	BE	BA	R1	R2	RS	BE	BA
Weakly Supervised															
QUERYSUM [34]	36.1	7.5	12.7	8.5	32.3	31.1	4.5	10.2	2.0	28.9	32.6	5.5	11.1	3.0	29.8
MARGESUM [35]	38.0	9.1	14.3	11.5	32.8	34.4	6.5	12.2	5.9	30.3	36.7	8.1	13.5	8.7	32.1
Supervised															
RAG-Sequence [19]	28.9	5.7	10.1	12.6	4.1	32.3	5.2	10.8	8.0	3.9	27.1	4.6	9.0	8.3	3.9
FiD [12]															
- BM25	42.4	11.3	16.5	21.4	38.1	38.8	8.4	14.2	15.7	35.0	41.4	10.8	16.1	20.0	36.8
- DPR	41.5	10.7	15.9	21.4	39.0	38.6	8.0	14.1	15.5	34.1	40.0	9.4	15.1	18.0	36.7
Zero-Shot Prompted															
NaiveRAG [9]															
- BM25	36.4	10.5	14.4	26.8	39.5	31.3	7.0	11.4	19.6	34.9	32.8	8.1	12.4	23.4	37.2
- DPR	37.1	10.4	14.7	27.3	39.5	31.7	7.1	11.4	18.3	34.8	33.7	8.4	12.6	21.8	37.1
Ours															
- BM25	43.1	11.0	16.5	25.9	40.7	37.9	7.3	13.1	19.9	34.5	39.4	8.8	14.3	22.1	37.2
- DPR	42.8	11.0	16.3	25.5	40.6	36.7	7.0	12.5	17.7	33.8	39.2	8.6	14.0	21.5	37.2
Few-Shot Prompted															
NaiveRAG [9]															
- BM25	41.7	11.9	16.5	24.4	41.7	35.2	7.8	12.8	17.1	35.5	37.5	9.1	14.2	20.2	37.7
- DPR	42.3	11.7	16.5	25.1	41.6	34.3	7.5	12.3	16.7	35.1	36.9	9.1	13.9	19.1	37.9
Ours															
- BM25	**45.8***	**13.4***	**18.6***	**29.2***	44.7	**41.7***	**9.6**	**15.6***	22.5	**37.3***	**43.6***	**11.3**	**16.7**	**26.1***	40.8
- DPR	45.1	12.9	18.3	28.6	**44.8***	41.1	9.4	15.2	**22.7***	36.8	42.9	10.7	16.2	24.7	**41.7***

7.3 Effect of Knowledge-Intensive Setup

To answer **RQ3**, we compare the performance of our best approach variant, which utilizes BM25, across original document sets (ORIGIN) and our three knowledge corpora. It is worth noting that the primary difference lies in the fact that ORIGIN contains only a limited set of manually curated relevant documents (around 40 documents) from original DUC datasets, whereas our knowledge corpora, such as CORPUS$_{Ext}$ and CORPUS$_{Aug}$, consists of millions of candidate documents. The results are shown in Table 3. We observe a significant decline in model performance when applied in the knowledge-intensive setup. For instance, when comparing ORIGIN and CORPUS$_{Aug}$, ROUGE-1 scores decrease by about 6 points, dropping from 49.8 to 43.6, and BERTScore scores decrease by about 6 points, dropping from 32.5 to 26.1. One possible explanation is that although both knowledge corpora offer substantial evidence for summarization, CORPUS$_{Aug}$ contains a considerable number of irrelevant documents. Consequently, retrieval models struggle to sift through this larger pool to identify the relatively small amount of relevant documents. This discrepancy underscores the

Table 3. Performance comparison of our approach variant with BM25 over the original document sets (ORIGIN) and our three knowledge corpora.

Corpus	ROUGE-1	ROUGE-2	ROUGE-SU4	BERTScore	BARTScore
ORIGIN	49.8	17.3	21.9	32.5	47.6
CORPUS$_{Int}$	45.8	13.4	18.6	29.2	44.7
CORPUS$_{Ext}$	41.7	9.6	15.6	22.5	37.3
CORPUS$_{Aug}$	43.6	11.3	16.7	26.1	40.8

significantly greater challenge posed by the knowledge-intensive setups compared to the traditional QFS, indicating a need for further research efforts to adapt retrieval models to these realistic scenarios.

8 Conclusion

This paper introduces a novel knowledge-intensive approach to QFS, aiming to address the limitations of traditional QFS setup that relies on the availability of relevant documents. This approach is tailored for practical scenarios involving highly specialized topics, eliminating the dependence on pre-existing document sets. The approach efficiently retrieves potentially relevant documents from a large-scale knowledge corpus based on the query. The summarization controller combines an LLM-based summarizer with a carefully tailored prompt, ensuring the quality of the generated summary. To assess the effectiveness of our proposed approach compared to strong baseline models, we introduce a specialized dataset, along with human-annotated relevance labels, to facilitate comprehensive retrieval and summarization evaluation. Extensive experiments demonstrate the superior performance of our method, particularly its ability to generate accurate summaries without relying on the availability of relevant documents initially. This underscores the versatility and practical applicability of our approach across diverse query scenarios, thereby contributing to advancements in QFS research.

Acknowledgements. This research was supported by the China Scholarship Council (CSC) under grant number 202008440470. The views expressed in the content belong solely to the authors and may not reflect the perspectives or endorsements of their respective employers or sponsors.

References

1. Asai, A., Gardner, M., et al.: Evidentiality-guided generation for knowledge-intensive NLP tasks. In: NAACL-HLT, pp. 2226–2243 (2022)
2. Brown, T.B., Mann, B., et al.: Language models are few-shot learners. In: NeurIPS (2020)

3. Chen, D., Fisch, A., et al.: Reading wikipedia to answer open-domain questions. In: ACL, pp. 1870–1879 (2017)
4. Craswell, N., Mitra, B., et al.: Overview of the TREC 2020 deep learning track. In: TREC, NIST Special Publication, vol. 1266 (2020)
5. Dang, H.T.: DUC 2005: evaluation of question-focused summarization systems. In: Proceedings of the Workshop on Task-Focused Summarization and Question Answering, pp. 48–55 (2006)
6. Fabbri, A., Li, I., et al.: Multi-news: a large-scale multi-document summarization dataset and abstractive hierarchical model. In: ACL, pp. 1074–1084 (2019)
7. Fan, A., Jernite, Y., et al.: ELI5: Long form question answering. In: ACL, pp. 3558–3567 (2019)
8. Fleiss, J.L.: Measuring nominal scale agreement among many raters. Psychol. Bull. **76**(5), 378 (1971)
9. Gao, Y., Xiong, Y., et al.: Retrieval-augmented generation for large language models: a survey. Arxiv (2023)
10. Giorgi, J., Soldaini, L., et al.: Open domain multi-document summarization: a comprehensive study of model brittleness under retrieval. In: EMNLP Findings, pp. 8177–8199. Association for Computational Linguistics (2023)
11. Huang, K.H., Singh, S., et al.: SWING: balancing coverage and faithfulness for dialogue summarization. In: EACL Findings, pp. 512–525 (2023)
12. Izacard, G., Grave, E.: Leveraging passage retrieval with generative models for open domain question answering. In: EACL, pp. 874–880 (2021)
13. Karpukhin, V., Oguz, B., et al.: Dense passage retrieval for open-domain question answering. In: EMNLP, pp. 6769–6781 (2020)
14. Krishna, K., Roy, A., et al.: Hurdles to progress in long-form question answering. In: NAACL-HLT, pp. 4940–4957 (2021)
15. Kulkarni, S., Chammas, S., et al.: AQuaMuSe: automatically generating datasets for query-based multi-document summarization. Arxiv **abs/2010.12694** (2020)
16. Laskar, M.T.R., Bari, M.S., et al.: A systematic study and comprehensive evaluation of ChatGPT on benchmark datasets. In: ACL Findings, pp. 431–469 (2023)
17. Laskar, M.T.R., Hoque, E., et al.: WSL-DS: weakly supervised learning with distant supervision for query focused multi-document abstractive summarization. In: COLING, pp. 5647–5654 (2020)
18. Lewis, M., Liu, Y., et al.: BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: ACL, pp. 7871–7880 (2020)
19. Lewis, P.S.H., Perez, E., et al.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: NeurIPS (2020)
20. Li, P., Wang, Z., et al.: Salience estimation via variational auto-encoders for multi-document summarization. In: AAAI, vol. 31 (2017)
21. Li, W., Zhang, X., et al.: Document-based question answering improves query-focused multi-document summarization. In: NLPCC, pp. 41–52 (2019)
22. Lin, C.Y.: ROUGE: a package for automatic evaluation of summaries. In: Text Summarization Branches Out, pp. 74–81 (2004)
23. Lin, J., Ma, X., et al.: Pyserini: a python toolkit for reproducible information retrieval research with sparse and dense representations. In: SIGIR, pp. 2356–2362 (2021)
24. Liu, Y., Wang, Z., et al.: QuerySum: a multi-document query-focused summarization dataset augmented with similar query clusters. In: AAAI, pp. 18725–18732 (2024)

25. Ma, C., Zhang, W.E., et al.: Multi-document summarization via deep learning techniques: a survey. ACM Comput. Surv. **55**(5), 102:1–102:37 (2023)
26. Min, S., Michael, J., et al.: AmbigQA: answering ambiguous open-domain questions. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 5783–5797 (2020)
27. Ouyang, L., Wu, J., et al.: Training language models to follow instructions with human feedback. In: NeurIPS (2022)
28. Petroni, F., Piktus, A., et al.: KILT: a benchmark for knowledge intensive language tasks. In: NAACL-HLT, pp. 2523–2544 (2021)
29. Raffel, C., Shazeer, N., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**, 140:1–140:67 (2020)
30. Robertson, S.E., Zaragoza, H.: The probabilistic relevance framework: BM25 and beyond. Found. Trends Inf. Retr. **3**(4), 333–389 (2009)
31. Shaham, U., Segal, E., et al.: SCROLLS: standardized CompaRison over long language sequences. In: EMNLP, pp. 12007–12021 (2022)
32. Stelmakh, I., Luan, Y., et al.: ASQA: factoid questions meet long-form answers. In: EMNLP, pp. 8273–8288 (2022)
33. Wang, A., Pang, R.Y., et al.: SQuALITY: building a long-document summarization dataset the hard way. In: EMNLP, pp. 1139–1156 (2022)
34. Xu, Y., Lapata, M.: Coarse-to-fine query focused multi-document summarization. In: EMNLP, pp. 3632–3645 (2020)
35. Xu, Y., Lapata, M.: Generating query focused summaries from query-free resources. In: ACL, pp. 6096–6109 (2021)
36. Yuan, W., Neubig, G., et al.: BARTScore: evaluating generated text as text generation. In: NeurIPS, pp. 27263–27277 (2021)
37. Zhang, T., Kishore, V., et al.: BERTScore: evaluating text generation with BERT. In: ICLR (2020)
38. Zhang, W., Aliannejadi, M., Pei, J., Yuan, Y., Huang, J.H., Kanoulas, E.: A comparative analysis of faithfulness metrics and humans in citation evaluation. In: LLM4Eval at SIGIR (2024)
39. Zhang, W., Aliannejadi, M., Yuan, Y., Pei, J., Huang, J.H., Kanoulas, E.: Towards fine-grained citation evaluation in generated text: a comparative analysis of faithfulness metrics. In: INLG (2024)
40. Zhang, W., Pal, V., Huang, J.H., Kanoulas, E., de Rijke, M.: QFMTS: generating query-focused summaries over multi-table inputs. In: ECAI (2024)
41. Zhang, W., Vakulenko, S., Rajapakse, T., Kanoulas, E.: Scaling up query-focused summarization to meet open-domain question answering. ArXiv preprint, abs/2112.07536 (2021)
42. Zhang, W., Vakulenko, S., Rajapakse, T., Xu, Y., Kanoulas, E.: Tackling query-focused summarization as a knowledge-intensive task: a pilot study. Gen-IR at SIGIR (2023)
43. Zhao, W.X., Liu, J., et al.: Dense text retrieval based on pretrained language models: a survey. ACM Trans. Inf. Syst. **42**(4), 1–60 (2024)
44. Zhong, M., Yin, D., et al.: QMSum: a new benchmark for query-based multi-domain meeting summarization. In: NAACL-HLT, pp. 5905–5921 (2021)
45. Zhou, Y., Shi, K., et al.: ODSum: new benchmarks for open domain multi-document summarization. Arxiv (2023)

Font Style Translation in Scene Text Images with CLIPstyler

Honghui Yuan and Keiji Yanai(✉)

The University of Electro-Communications, Chofu, Tokyo, Japan
{yuan-h,yanai}@mm.inf.uec.ac.jp

Abstract. Scene text editing is widely used in various fields, such as poster design and correcting spelling mistakes in the image. Editing text in images is a challenging task that requires accurately and naturally integrating text within complex backgrounds. Existing methods have achieved changing the text content with the target text without altering the style of text and the background of the image. However, arbitrary style transformation of the text region in the image has not been achieved. To address this issue, we propose a new framework named FontCLIPstyler, which enables the style transformation of text in scene text images using prompts. The proposed method mainly comprises two networks: MaskNet, which extracts mask images of the text region in images, and StyleNet, which performs the generation of stylized images. In addition, we also propose a new loss function named Text-aware Loss, which can guide the StyleNet network in transferring style features to the text region without changing the background. Through extensive experiments and ablation studies, we have demonstrated the effectiveness of our method in scene text style transformation. The experimental results show that our approach can successfully transfer the semantic style from the input prompt to the text region of the image, and create naturally stylized scene text while keeping the readability of the text and the background invariant.

Keywords: Image style transfer · Font translation · Scene text images · Arbitrary style · CLIPstyler

1 Introduction

In recent years, the development of deep learning has significantly enhanced the convenience of image editing. Many studies have achieved significant results on scene text editing using GAN(Generative Adversarial Networks) [9] and the diffusion model [26]. The previous methods of scene text editing focused on replacing the text content within the image while preserving the background and the text style (color, texture, font, etc.). However, these methods are limited to text content replacement and cannot transform text style arbitrarily. Therefore, as shown in Fig. 1, we propose a task focused on scene text style transformation, aiming to translate the style of the text region without modifying the text content and the

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 105–121, 2025.
https://doi.org/10.1007/978-3-031-78495-8_7

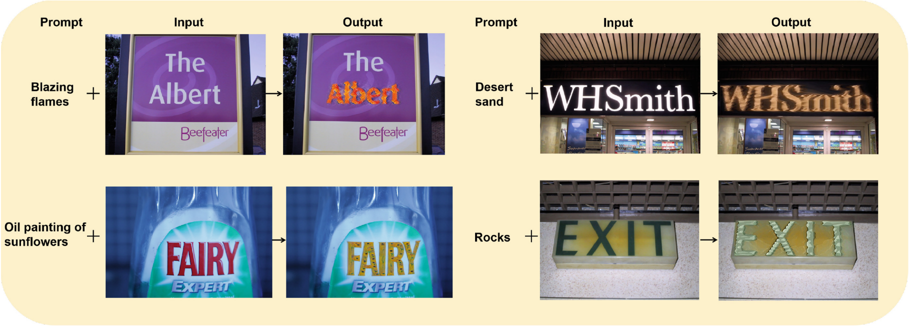


Fig. 1. The proposed method can transfer the style of text regions in images with prompts.

background in the image. In general, scene text editing is divided into three subtasks including background restoration, text conversion, and image re-synthesize. However, this process relies on using the original image as a style reference, ensuring that the generated image maintains the same text style and background as the original image. Recently, with the advance of the diffusion [26] model, many methods based on the diffusion model can generate natural and high-quality scene text images. However, when dealing with editing the text region of the image, these models could only transfer text style based on the style of other text in the same image, thus lacking arbitrary style transformation ability.

Image style transformation usually refers to generating new images by integrating the content of one image with the style of another image. A lot of methods [5,38] have successfully generated attractive results with the GAN and transformer [32] models. However, sometimes it is hard to find the desired style images as the style reference image of style transfer. Recently, the CLIP [25] model, a multi-modal model of language and images, has been utilized to guide image generation through prompts. CLIPstyler [20] utilizes a lightweight CNN network and the CLIP model to achieve the image style transformation through the simple text description, without the need for style reference images. Also, the main content of the original image can be effectively maintained unchanged while applying the style transformation. Different from the style transformation of normal images, the style transformation of text images is more complex because it is necessary to ensure the readability of the text while transferring the style features to the text. Many studies [34,37] have focused on the style transformation of text images, which usually utilize the network that uses text images to learn the text structure so that the content of the text will be maintained. However, these studies are usually limited to images of single characters, and cannot handle images of multiple characters such as words.

Therefore, we propose a new framework based on CLIPstyler [20] named FontCLIPstyler, to achieve the style transformation of text region in the scene text image without changing the image background and text content. The proposed method enables style transformation using prompts and enables arbitrary style transfer. Our main contributions are as follows:

1. We propose a style transfer framework for scene text images that can effectively transfer the style of the text region in the image while keeping the background and text content unchanged.
2. Our method uses prompts to control the style without the style reference image, achieving arbitrary style transformation of the scene text image.
3. The experiments have proved that our method can generate visually attractive styled scene text and successfully achieve the scene text style transfer task.

2 Related Work

2.1 Scene Text Editing

Scene text editing has made significant progress in replacing text in the original image with another text while preserving the style of the text. STEFANN [28] designed two networks for font structure transformation and color transformation. However, it is replacing one character of text at a time and is unable to vary the number of characters different from the original text. SRNet [36] achieved text replacement using three sub-networks: background restoration, text conversion, and image re-synthesis. SwapText [39] incorporates a TPS(Thin-Plate-Spline) module and utilizes spatial points to geometrically transform text based on SRNet. SimAN [22] implemented similarity-aware normalization and uses a self-supervised learning method to train the network. Based on StyleGAN [18], TextStyleBrush [19] integrates style vectors of the text image into the generator to guide the generation of final images. Mostel [24] added additional stroke-level information to the network and used synthetic and real-world data to significantly improve scene text editing performance. However, these methods require the style reference image to achieve style transformation of text. Recently, the diffusion model [26] had great success in image generation and editing. DiffSTE [16], DiffUTE [3], GlyphDraw [23] GlyphControl [42], TextDiffuser [4], and some other methods based on the diffusion model have achieved high-quality scene text generation and editing. However, they do not offer control over the style of the text. Different from the methods mentioned above, our method does not require the style reference image for style transfer and allows users to specify the scene text style by prompts.

2.2 Style Transfer

Image style transformation aims to transfer the style from a reference image to a target image. Neural image style transfer [8] utilizes a CNN-based network to achieve image style transfer based on style images. AdaIN [13] using adaptive instance normalization to align the mean and variance of the content image's features with those of the style image, enabling arbitrary style transformations. In recent years, based on GAN [9] and Tansformer [32], methods such as StyleGAN [18], StyTr2 [5] have achieved significant success in generating high-quality style images. However, these methods require the style reference image to get the

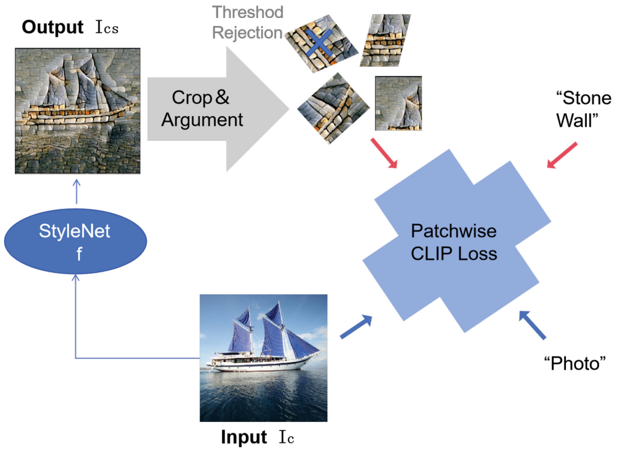


Fig. 2. Overview of CLIPstyler.

style features. By learning models from a dataset consisting of 4 billion pairs of images and text, Recent research CLIP [25] can improve zero-shot performance for many downstream tasks. As shown in Fig. 2, Based on the CLIP [25] model, CLIPstyler [20] has solved the problem that needs the reference style image to transfer the style of images and allowing arbitrary style transformations with prompts. Sem-CS [17] and Gen-Art [43] used semantic segmentation to solve the over-stylization problem of the foreground portion in CLIPstyler [20]. However, these methods perform style transformations on the entire image and are unable to transfer the style to specific parts of the image.

Text image style transformation methods have been widely researched due to the success of image style transfer. MCGAN [2] focused on the English alphabet, generating the remaining alphabet in the same style based on a few alphabet examples. TETGAN [40] enables style transfer from one character to others via a stylization subnetwork and a de-stylization subnetwork. FETGAN [21] enables the generation of new characters in the same style with only a few artistic characters, supporting both the English alphabet and more complex Chinese characters. Typography with Decor [35] proposed a novel deep-learning network to achieve the style transfer of characters that include decoration. Shape Matching-GAN [41] allows converting text styles by using just one style reference image. Multi-Style Shape Matching GAN [12] used a single model to achieve multiple styles generation of text images based on Shape MatchingGAN. However, these methods depend on style images or other text images as the style reference images. More recently, Word as Image [14], CLIPFont [29], DS-Fusion [31], Zero-shot Font Style Transfer [15] has achieved font style transfer using the prompt without the need for style images. However, these methods are limited to single-character images and struggle with generating satisfactory results for multiple characters, such as words. Our method is capable of transferring the style of text regions in scene text images through prompts and can preserve the readability of text unrestricted to the number of characters.

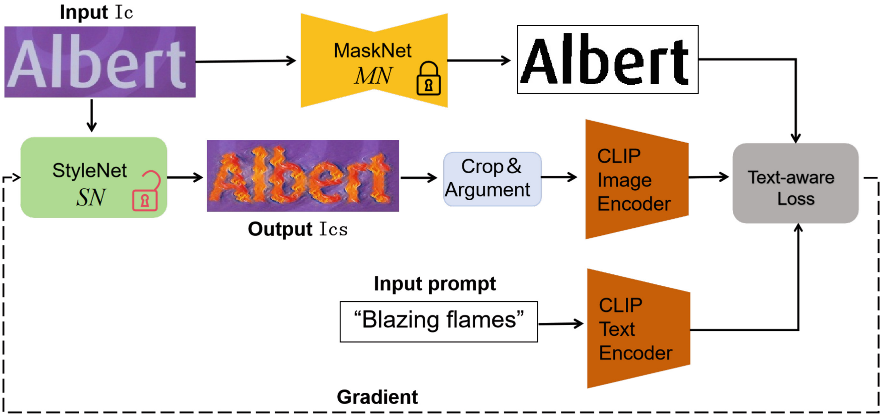


Fig. 3. The framework of our method FontCLIPstyler.

3 Methodology

In this study, we propose a new framework based on CLIPstyler [20] named FontCLIPstyler to achieve style transformation of the text region in scene images. An overview of our proposed method is shown in Fig. 3. The network mainly consists of two networks, the MaskNet network (MN) that extracts the mask image of the text region in the scene text image, and the StyleNet network(SN) that conducts style transformation of the input image. By using the pre-trained text-image embedding model CLIP [25] and the Text-aware Loss proposed in this study, the parameters of the network SN are optimized to transfer the semantic style from input prompts to the generated image.

3.1 Basic Framework CLIPstyler

Firstly, we will start with an introduction to CLIPstyler, which has been used as the base model for our method. As shown in Fig. 2, CLIPstyler aims to transfer the semantic style features from the input prompt to the input image. Since the style is expressed in the form of natural language, it does not require the style image as a reference to get style features. Without the constraints of reference style images, which are sometimes difficult to obtain, arbitrary style transformations can be realized through imaginative prompts. Specifically, the input image I_c is fed into StyleNet f which is the encoder-decoder CNN, and the parameters of f are optimized by Patchwise CLIPLoss to transfer style features from prompts to the image and generate the stylized image I_{cs}. When calculating the loss function, the generated images are randomly cropped and augmented to achieve more vivid and diverse textures. The loss function used in the CLIPstyler can be formed as follows.

$$L_{total} = \lambda_d L_{dir} + \lambda_p L_{patch} + \lambda_c L_c + \lambda_{tv} L_{tv} \tag{1}$$

The Directional CLIPLoss(L_{dir}) proposed by StyleGAN-NADA [7] be applied to guide the output image rendering with the semantic style of the target prompt. As follows, Directional CLIPLoss encodes the input image I_c, input prompt t_{style}, output image $f(I_c)$ and content prompt t_{src} by CLIP Encoder. By aligning the direction between these features, the generated images have the same semantic style as the input prompts.

$$\begin{aligned} \Delta T &= E_T(t_{style}) - E_T(t_{src}), \\ \Delta I &= E_I(f(I_c)) - E_I(I_c), \\ L_{dir} &= 1 - \frac{\Delta I \cdot \Delta T}{|\Delta I|\,|\Delta T|} \end{aligned} \tag{2}$$

Patch CLIPLoss(L_{patch}) proposed by CLIPstyler [20] using the randomly cropped patches $\hat{I}^i_{cs}$ to calculate Directional CLIPLoss instead of the entire generated image to get more high-quality images. In addition, random geometrical augmentation was applied to the cropped patches before calculating the Directional CLIPLoss and achieved more vivid and diverse textures. Furthermore, to prevent over-stylization of the image, a specific threshold τ is set to reject patches that are below this threshold. To retain the content of the original image in the generated image, use the VGG-19 network to calculate the content loss L_c. Furthermore, CLIPstyler also utilized total variation regularization loss (L_{tv}), which reduces the side artifacts caused by irregular pixels in the image. The calculation of Patch CLIPLoss is as follows.

$$\begin{aligned} \Delta T &= E_T(t_{style}) - E_T(t_{src}), \\ \Delta I &= E_I(aug(\hat{I}^i_{cs})) - E_I(I_c), \\ L^i_{patch} &= 1 - \frac{\Delta I \cdot \Delta T}{|\Delta I|\,|\Delta T|}, \\ L_{patch} &= \frac{1}{N}\sum_i^n R(l^i_{patch}, \tau) \\ where\ R(s,\tau) &= \begin{cases} 0, & \text{if } s \le \tau \\ s, & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

The calculation of the total variation regularization loss L_{tv} is as follows. This function calculates the sum of squared gradient differences of the pixel values in horizontal, vertical, and two diagonal directions for the input image x of size $H * W$ across the three channels c.

$$\begin{aligned} L_{tv} = &\sum_{c=1}^{3}\sum_{i=1}^{H}\sum_{j=1}^{W-1}(x_{i,j,c} - x_{i,j+1,c})^2 + \sum_{c=1}^{3}\sum_{i=1}^{H-1}\sum_{j=1}^{W}(x_{i,j,c} - x_{i+1,j,c})^2. \\ &+ \sum_{c=1}^{3}\sum_{i=1}^{H-1}\sum_{j=1}^{W-1}(x_{i+1,j,c} - x_{i,j+1,c})^2 + \sum_{c=1}^{3}\sum_{i=1}^{H-1}\sum_{j=1}^{W-1}(x_{i,j,c} - x_{i+1,j+1,c})^2 \end{aligned} \tag{4}$$

3.2 FontCLIPstyler

CLIPstyler [20] addresses the problem that needs style images as references in image style transformation, making the process more convenient. By using prompts as guidance, the model has enabled arbitrary style transformations. However, CLIPstyler applies style transformations to the entire image and cannot target specific regions within the image for style transformation. To overcome this limitation, we propose the FontCLIPstyler framework based on CLIPstyler, which focuses on the style transformation of text region in the scene text image. In the following, we will provide a detailed description of the proposed framework and the loss function.

MaskNet and StyleNet. Our framework is mainly composed of two networks: MaskNet and StyleNet. For editing an image in a specific area without impacting other regions, a straightforward and effective approach is to employ the mask image to delineate the targeted area. Thus, we introduce the MaskNet network designed to extract the mask image of the text region in scene images. The mask image serves as guidance for the method, enabling the style transfer to the text region of a scene text image while keeping the image background and text content unchanged. Specifically, because of the high computational efficiency and real-time image processing capability of U-Net [27], it is used as the backbone of the proposed MaskNet network. We train the MaskNet with real-world scene text images, and during the style transfer process, it is frozen to generate the mask image of the specified text region. In addition, with reference to CLIPStyler, we employ a CNN encoder-decoder network StyleNet for the style transfer. During training, the parameters of StyleNet are optimized with our proposed loss function Text-aware Loss, allowing for the generation of styled scene text images with the input prompt.

Text-Aware Loss. We propose a new loss function named Text-aware Loss that could control the StyleNet network to realize style transformation into the text region in the image while maintaining the background invariably and the readability of the text. Our proposed loss is mainly composed of three parts: Distance loss, TextPatch CLIPLoss, and Background reconstruction loss. Distance Transform loss [1] allows style transformation within a limited region based on distance transformation of the input image. Therefore, to transform the style in the text region of the scene text image, we utilize Distance Transform loss in this study. The loss function can be defined as follows. Specifically, a distance transform map I_d is created using the mask image obtained from the proposed MaskNet network. The distance transform assigns to each pixel $I_d^{(i,j)}$ a value that represents the distance to the nearest target region pixel. Using the Euclidean distance metric, this transformation calculates the distance between each pixel of the distance transform map $p_{i,j}$ and the pixel of mask image $p_{x,y}$. Pixels that are part of the target region are assigned a value of zero. As the pixels get further from the target, their assigned values, which represent their distance from the nearest target pixel, increase. Then, the input image I_c and stylized output image I_{sty} from StyleNet will multiplied by I_d respectively, and the mean squared error is calculated.

$$I_d^{(i,j)} = \min_{x,y \in mask} ||p_{i,j} - p_{x,y}||_2 \tag{5}$$

$$L_{distance} = \frac{1}{2}\sum_{i,j}(I_c^{(i,j)} \cdot I_d^{(i,j)} - I_{sty}^{(i,j)} \cdot I_d^{(i,j)})^2 \tag{6}$$

When utilizing prompts to control the style transfer, it is necessary to use Patch CLIPLoss to reflect the semantic style of the input prompt into the image. However, in CLIPstyler, Patch CLIPLoss randomly crops patches from the image to be stylized and tends to include background areas that do not require style transformation for our task. As a result, style features are reflected in the background resulting in background changes. To solve this problem, we propose a new loss function, TextPatch CLIPLoss. Specifically, the patches of the background region $I^i_{b\_patch}$ for the input image I_c are cropped using the mask image($Mask$) obtained from MaskNet(MN). The cosine similarity (sim) with the patches of the generated image and background region image $I^i_{b\_patch}$ is calculated, as a standard for determining the regions. Specifically, we utilize the image encoder of CLIP(E_I) to encode the generated image $\hat{I}^i_{sty}$ and the background region image $I^i_{b\_patch}$ of N patches. Then, we set a threshold μ to determine whether the generated image belongs to the background or text region. Patches with significantly different similarities are determined to belong to the text region $\hat{I}^i_{sty\_t}$. Then Patch CLIPLoss is calculated as CLIPstyler for patches belonging to the text region to render the style features using CLIP text encoder (E_T) and CLIP image encoder (E_I). In addition, when calculating Patch CLIPLoss, CLIPstyler sets a threshold τ to prevent the image from being over-stylization and get a better result in preserving the main content of images. However, it is more difficult to render style features into text regions in scene text images than usual images because the font typically consists of elongated structures, and showing visible and attractive style features is more difficult. Therefore, the threshold used to avoid over-stylization makes it difficult to reflect style features into the text region in scene images. To better render the style features into the text, we used all the patches in our loss function. L_{patch} is calculated as follows:

$$\begin{aligned} &Mask = MN(I_c), \\ &I^i_{b\_patch} = crop((1 - Mask) \odot I_c), \\ &sim = 1 - \frac{E_I(\hat{I}^i_{sty}) \cdot (\frac{1}{N}\sum_i^n E_I(I^i_{b\_patch}))}{\left|E_I(\hat{I}^i_{sty})\right| \left|\frac{1}{N}\sum_i^n E_I(I^i_{b\_patch})\right|}, \\ &\hat{I}^i_{sty} = \begin{cases} \hat{I}^i_{sty\_b} & if \quad sim < \mu \\ \hat{I}^i_{sty\_t} & otherwise \end{cases} \end{aligned} \tag{7}$$

$$\begin{aligned} \Delta T &= E_T(t_{style}) - E_T(t_{src}), \\ \Delta I &= E_I(aug(\hat{I}^i_{sty\_t})) - E_I(I_c), \\ L^i_{patch} &= 1 - \frac{\Delta I \cdot \Delta T}{|\Delta I|\,|\Delta T|}, \\ L_{patch} &= \frac{1}{N}\sum_i^n l^i_{patch} \end{aligned} \tag{8}$$

The content loss L_c is used in CLIPStyler to retain the main contents of the generated image unchanged from the original image. However, it will lead to the text style of the original image being reflected in the generated image. Therefore, to reduce the influence of the text style from the original image and make sure the content of the background is unchanged, instead of using the content loss L_c, we utilized a background reconstruction loss L_{recon}. Specifically, VGG19 loss is calculated as follows between the N patches of the generated image belonging to the background region $\hat{I}^i_{sty\_b}$ and the cropped patches from the input image I^i_c. Here F_{4_2} and F_{5_2} represent the convolution layer conv4_2 and conv5_2 of the VGG19 network, respectively.

$$\begin{aligned} L^i_{recon} &= \left\| F_{4\_2}(\hat{I}^i_{sty\_b}) - F_{4\_2}(I^i_c) \right\|^2_2 \\ &\quad + \left\| F_{5\_2}(\hat{I}^i_{sty\_b}) - F_{5\_2}(I^i_c) \right\|^2_2 \\ L_{recon} &= \frac{1}{N} \sum_i^n l^i_{recon} \end{aligned} \tag{9}$$

The loss function of the proposed Text-aware Loss L_{ta} is as follows:

$$L_{ta} = \lambda_d L_{distance} + \lambda_p L_{patch} + \lambda_r L_{recon} \tag{10}$$

where λ represents the weight of each loss function.

We also utilized the total variation regularization loss L_{tv} to reduce the side artifacts caused by irregular pixels in the image as CLIPstyler. Thus, the total loss function is as follows:

$$L_{total} = L_{ta} + \lambda_{tv} L_{tv} \tag{11}$$

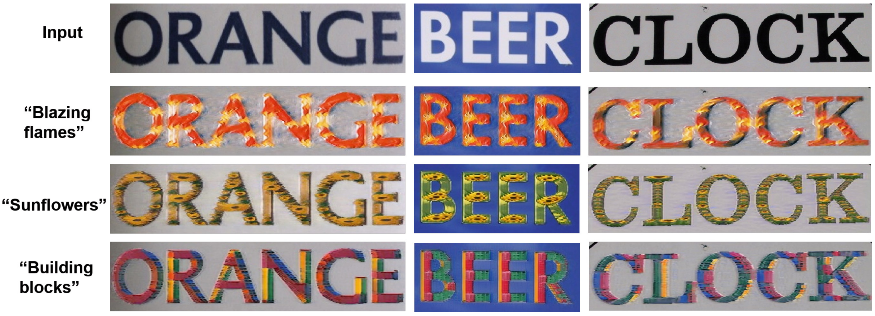


Fig. 4. The results of different style transformations on the same scene text images using the proposed method.

4 Experiment

Implementation Details. The proposed MaskNet network was trained using 2000 real-world scene text images collected from Mostel [24] and we randomly selected 200 other synthetic and real-world images to test the MaskNet. The network of MaskNet consists of four downsample and upsample layers, with 64, 128, 256, and 512 channels respectively, followed by max-pooling. And two bottleneck layers with 1024 channels, the final layer is a sigmoid function. When conducting style transformation, MaskNet is frozen and the StyleNet network is optimized with the loss function without training. We conducted experiments with real scene text images that were randomly selected from the COCOText V2.0 dataset [33]. As for the network of StyleNet, We utilize a lightweight U-net [27] architecture featuring three down-sampling and three up-sampling layers. The channel sizes for each down-sampling layer are 16, 32, and 64, and the sigmoid function at the last layer. The input scene text images are converted to 512×512 size and the final output result will be resized to the original size. We set λ_d, λ_p, λ_r and λ_{tv} to 1×10^2, 9×10^3, 150 and 2×10^{-3}. The model is trained using a learning rate of 5×10^{-4} and Adam optimizer. Training iteration is set to 500 and the learning rate is halved every 100 iterations. We used a single NVIDIA TITAN RTX to train the model, and the training time per image was approximately 90 to 120 s.

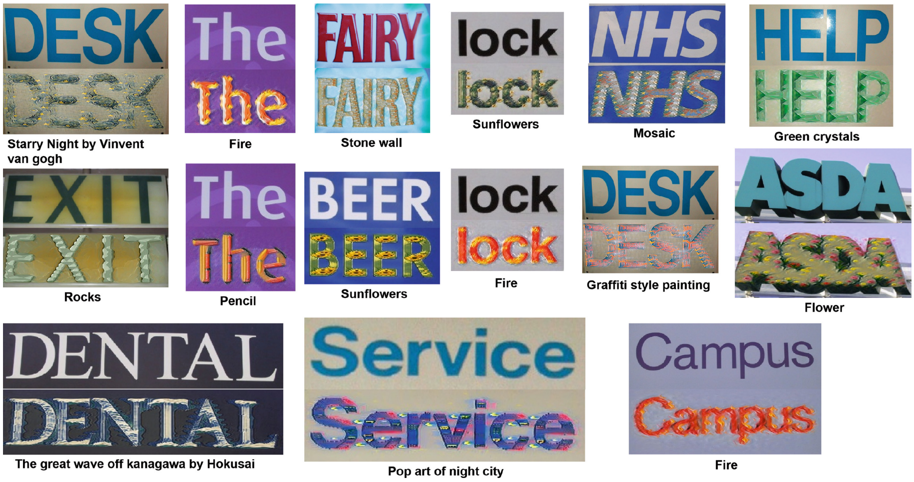


Fig. 5. The results of the proposed method with various prompts and different scene text images.

4.1 Evaluation

We conducted experiments under various conditions to verify the effectiveness of the proposed method in generating stylized scene text images. Specifically, we

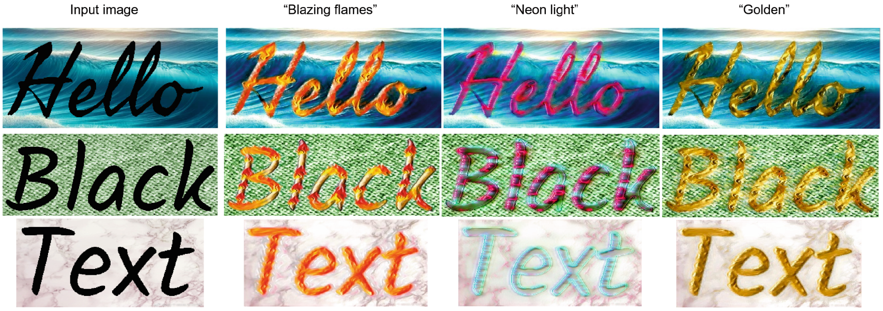


Fig. 6. The results of style transformation using the proposed method for synthesized scene text images.

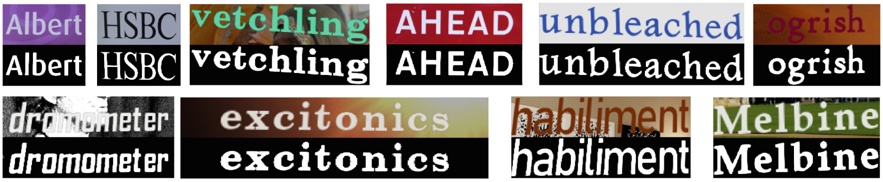


Fig. 7. The results of mask images using the proposed MaskNet.

have experimented with our method in various prompts and different scene text images. As shown in Fig. 4, we confirm the ability of our method in scene text style transfer by using the same image with different prompts. The proposed method successfully reflects the semantic style specified by the prompt to the text region, without changing the background and maintaining the readability of the original text content. Additionally, Fig. 5 shows more results from our experiments. Regardless of the length as well as the size of the text, our method effectively rendered the style features into the text. These results demonstrate the effectiveness of our proposed method in arbitrary style transformation of scene text images using prompts. To confirm that the proposed method can achieve style transformation of text even on complex backgrounds, experiments were conducted using high-resolution synthetic scene text images. As shown in Fig. 6, our method generated high-quality stylized text while preserving the fine texture of the background. Figure 7 shows the test results of our MaskNet. The results demonstrate that our network can effectively extract the masks of the text parts in various scenes.

4.2 Ablation Studies

We conducted ablation studies to verify the effectiveness of each component of the proposed Text-aware Loss. As shown in Fig. 8, without Distance Transform loss, the style features render over the entire image, and the text content is unidentifiable. The result has failed in the style transformation to the text area.

If TextPatch CLIPLoss is not applied, the semantic style from the input prompts is not reflected well in the image. When background reconstruction loss is not utilized, the featuress of the background will affect the text causing the boundary between text and background regions to become unclear and making text content difficult to identify. The model with all loss functions was able to transfer style features in the text region while preserving the background and text content. The quantitative evaluation results are shown in Table 1. Without the $L_{distance}$ and L_{recon} results in a decrease in each score, and without L_{patch} leads to poor qualitative results. Our full model achieves the highest scores in both NIMA [30] and CLIP scores [10], with the remaining scores being in second place. This demonstrates the effectiveness of each component of our model in generating aesthetic images and images that are consistent with the given prompts.

Table 1. Quantitative evaluation results for ablation studies.

	DISTS↓	NIMA↑	LPIPS↓	FID↓	CLIP SCORE↑
w/o Ldistance	0.4997	4.5776	0.6082	910.50	0.2428
w/o Lpatch	**0.3132**	4.4650	**0.5489**	**318.72**	0.2149
w/o Lrecon	0.4196	4.7785	0.5950	713.20	0.2536
Full model	0.4075	**4.9550**	0.5846	485.00	**0.2583**

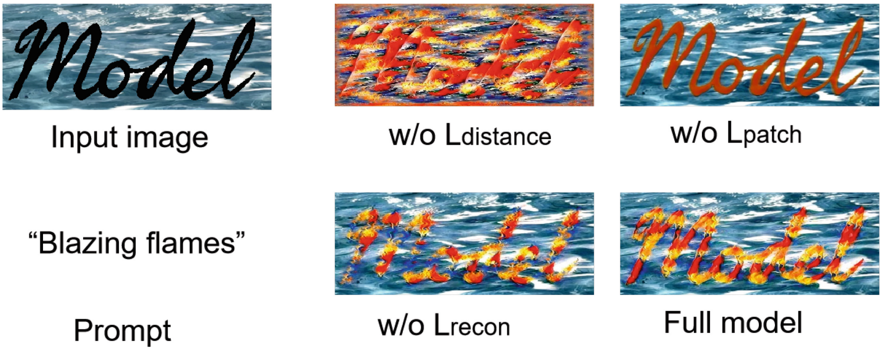


Fig. 8. Results of ablation studies.

Table 2. Quantitative evaluation between our method and previous methods.

	DISTS↓	NIMA↑	LPIPS↓	FID↓	CLIP SCORE↑
CLIPstyler	0.3901	4.6776	0.7171	**372.40**	0.1848
Sem-CS	0.3838	4.7322	0.6984	460.79	0.2065
Ours	**0.3324**	**4.8632**	**0.6667**	445.55	**0.2101**

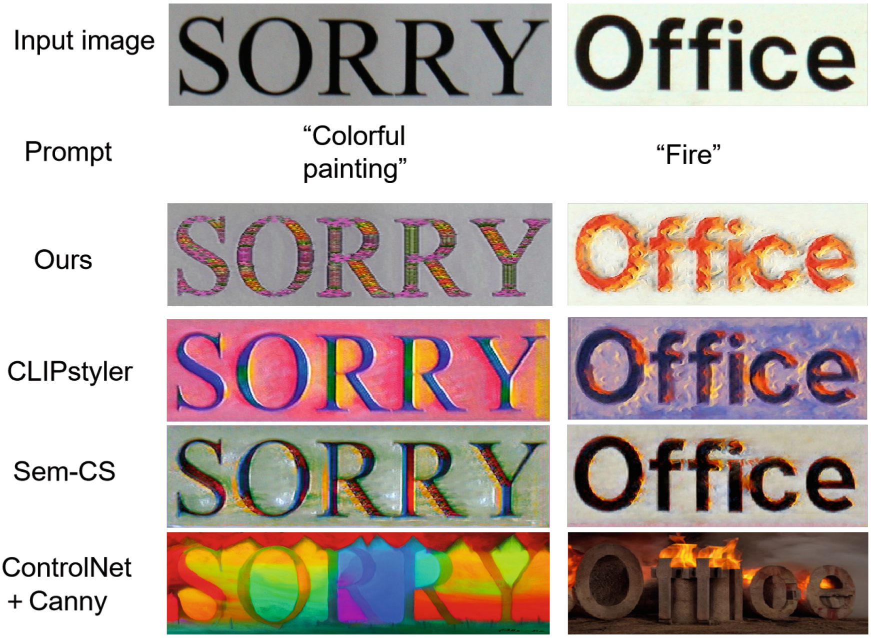


Fig. 9. The comparison results with other methods.

4.3 Comparisons with Previous Methods

Existing methods of scene text editing address the conversion of text content and do not allow arbitrary style transformations of text. Therefore, in addition to the base method CLIPstyler, we compare our method with the style transfer methods that are closer to the purpose of our study. Sem-CS [17] and ControlNet [44] achieved image style transformation using prompts and more focus on the main part of the image rather than the whole image. The results of the comparison are shown in Fig. 9. The results of CLIPstyler show that the style features were reflected throughout the entire image, resulting in the whole image change. Even though the results of Sem-CS reflect the style mainly on the text parts, style features tend to be around the text not inside the text, and make the style features not noticeable. The results of ControlNet reflected a noticeable style transformation, but the background changed significantly and the readability of the text declined. All these methods changed the background while achieving style transformation. Compared to previous methods, the proposed method achieved the best results and is successful in transforming the style into the text region without changing the background and readability of the text.

Regarding qualitative evaluation, we used DISTS [6] and NIMA [30] scores to evaluate our method and compare it with previous methods. DISTS utilizes a texture resampling full-reference image quality model that matches human evaluations of image quality. NIMA proposes a deep CNN that can predict the distribution of image evaluation on human opinions from a direct view (technical perspective) and attractiveness (aesthetic perspective), thereby evaluating the images in terms of human perception. We evaluated 100 images of scene text style transformations obtained from our model and other methods. The results are shown in Table 2. We achieved the best score in DISTS and demonstrated that our method realized style transformation in the text region while the generated image maintained the consistency of structure and texture information with the original image. We also obtained a better score in NIMA. Previous methods modified the entire image which could ensure the integrity of the image, under the condition of only changing a part of the image, our approach is still capable of generating naturally stylized scene text images with an aesthetic perspective. We also evaluate the LPIPS and FID [11] to value the similarity of the generated stylized image with the style image obtained by stable-diffusion-2-1-base with the same prompt. Moreover, We utilize the CLIP score to measure the similarity between generated images and prompts. Although our method only stylized the text region, while the other methods stylized the entire image, we achieved the best score in LPIPS, second only to CLIPstyler in FID. We obtained the best score in CLIP score, proving that our results were closest to the prompts.

5 Conclusion

In this paper, we proposed a new framework FontCLIPstyler to achieve the style transformation of scene text images. The proposed method does not require the style reference images and achieves arbitrary style transformation of text regions in the scene image using prompts. With the Text-aware Loss and MaskNet network, the proposed method solved the problem of CLIPstyler's inability to transform the style to a specific region in the image. The experimental results confirmed that our method could generate visually attractive stylized scene text while preserving the image background and text content. Our work could offer assistance in editing images like posters and designing more attractive artwork. Although this research realized the scene text style transformation, it is currently applicable only to the English alphabet, and can not transfer with more complicated text like Chinese characters or Japanese Kanji letters, since the MaskNet network only supports alphabet characters. In the future, we will continue to work on the realization of scene text style transformation in other languages.

References

1. Atarsaikhan, G., Iwana, B.K., Uchida, S.: Contained neural style transfer for decorated logo generation. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pp. 317–322 (2018)
2. Azadi, S., Fisher, M., Kim, V.G., Wang, Z., Shechtman, E., Darrell, T.: Multi-content GAN for few-shot font style transfer. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 7564–7573 (2018)
3. Chen, H., et al.: DiffUTE: universal text editing diffusion model. In: Advances in Neural Information Processing Systems, vol. 36 (2024)

4. Chen, J., Huang, Y., Lv, T., Cui, L., Chen, Q., Wei, F.: TextDiffuser: diffusion models as text painters. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
5. Deng, Y., et al.: StyTr2: image style transfer with transformers. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 11326–11336 (2022)
6. Ding, K., Ma, K., Wang, S., Simoncelli, E.P.: Image quality assessment: unifying structure and texture similarity. IEEE Trans. Pattern Anal. Mach. Intell. **44**(5), 2567–2581 (2020)
7. Gal, R., Patashnik, O., Maron, H., Bermano, A.H., Chechik, G., Cohen-Or, D.: StyleGAN-nada: CLIP-guided domain adaptation of image generators. ACM Trans. Graph. (TOG) **41**(4), 1–13 (2022)
8. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 2414–2423 (2016)
9. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, vol. 27 (2014)
10. Hessel, J., Holtzman, A., Forbes, M., Bras, R.L., Choi, Y.: CLIPScore: a reference-free evaluation metric for image captioning. arXiv preprint arXiv:2104.08718 (2021)
11. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: GANs trained by a two time-scale update rule converge to a local nash equilibrium. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
12. Honghui, Y., Keiji, Y.: Multi-style shape matching GAN for text images. IEICE Trans. Inf. Syst. **E107-D**, 505–514 (2024)
13. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 1501–1510 (2017)
14. Iluz, S., Vinker, Y., Hertz, A., Berio, D., Cohen-Or, D., Shamir, A.: Word-as-image for semantic typography. ACM Trans. Graph. (TOG) **42**(4), 1–11 (2023)
15. Izumi, K., Yanai, K.: Zero-shot font style transfer with a differentiable renderer. In: Proceedings of the 4th ACM International Conference on Multimedia in Asia, pp. 1–5 (2022)
16. Ji, J., et al.: Improving diffusion models for scene text editing with dual encoders. arXiv preprint arXiv:2304.05568 (2023)
17. Kamra, C.G., Mastan, I.D., Gupta, D.: Sem-CS: semantic CLIPStyler for text-based image style transfer. In: IEEE International Conference on Image Processing (ICIP), pp. 395–399 (2023)
18. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 4401–4410 (2019)
19. Krishnan, P., Kovvuri, R., Pang, G., Vassilev, B., Hassner, T.: TextStyleBrush: transfer of text aesthetics from a single example. IEEE Trans. Pattern Anal. Mach. Intell. (2023)
20. Kwon, G., Ye, J.C.: CLIPStyler: image style transfer with a single text condition. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 18062–18071 (2022)
21. Li, W., He, Y., Qi, Y., Li, Z., Tang, Y.: Fet-GAN: font and effect transfer via k-shot adaptive instance normalization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 1717–1724 (2020)
22. Luo, C., Jin, L., Chen, J.: SimAN: exploring self-supervised representation learning of scene text via similarity-aware normalization. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 1039–1048 (2022)

23. Ma, J., et al.: GlyphDraw: learning to draw Chinese characters in image synthesis models coherently. arXiv preprint arXiv:2303.17870 (2023)
24. Qu, Y., Tan, Q., Xie, H., Xu, J., Wang, Y., Zhang, Y.: Exploring stroke-level modifications for scene text editing. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 2119–2127 (2023)
25. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763 (2021)
26. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 10684–10695 (2022)
27. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 234–241 (2015)
28. Roy, P., Bhattacharya, S., Ghosh, S., Pal, U.: STEFANN: scene text editor using font adaptive neural network. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 13228–13237 (2020)
29. Song, Y., Zhang, Y.: CLIPFont: text guided vector wordart generation. In: British Machine Vision Conference. BMVA Press (2022). https://bmvc2022.mpi-inf.mpg.de/0543.pdf
30. Talebi, H., Milanfar, P.: NIMA: neural image assessment. IEEE Trans. Image Process. **27**(8), 3998–4011 (2018)
31. Tanveer, M., Wang, Y., Mahdavi-Amiri, A., Zhang, H.: DS-fusion: artistic typography via discriminated and stylized diffusion. In: Proceedings of IEEE International Conference on Computer Vision, pp. 374–384 (2023)
32. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
33. Veit, A., Matera, T., Neumann, L., Matas, J., Belongie, S.: Coco-text: dataset and benchmark for text detection and recognition in natural images. arXiv preprint arXiv:1601.07140 (2016)
34. Wang, C., Zhou, M., Ge, T., Jiang, Y., Bao, H., Xu, W.: CF-Font: content fusion for few-shot font generation. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 1858–1867 (2023)
35. Wang, W., Liu, J., Yang, S., Guo, Z.: Typography with decor: intelligent text style transfer. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 5889–5897 (2019)
36. Wu, L., et al.: Editing text in the wild. In: Proceedings of ACM International Conference Multimedia, pp. 1500–1508 (2019)
37. Xie, Y., Chen, X., Sun, L., Lu, Y.: DG-Font: deformable generative networks for unsupervised font generation. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 5130–5140 (2021)
38. Xu, W., Long, C., Wang, R., Wang, G.: DRB-GAN: a dynamic resblock generative adversarial network for artistic style transfer. In: Proceedings of IEEE International Conference on Computer Vision, pp. 6383–6392 (2021)
39. Yang, Q., Huang, J., Lin, W.: SwapText: image based texts transfer in scenes. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 14700–14709 (2020)
40. Yang, S., Liu, J., Wang, W., Guo, Z.: TET-GAN: text effects transfer via stylization and destylization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 1238–1245 (2019)

41. Yang, S., Wang, Z., Wang, Z., Xu, N., Liu, J., Guo, Z.: Controllable artistic text style transfer via shape-matching GAN. In: Proceedings of IEEE Computer Vision and Pattern Recognition, pp. 4442–4451 (2019)
42. Yang, Y., et al.: GlyphControl: glyph conditional control for visual text generation. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
43. Yang, Z., Song, H., Wu, Q.: Generative artisan: a semantic-aware and controllable clipstyler. arXiv preprint arXiv:2207.11598 (2022)
44. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of IEEE International Conference on Computer Vision, pp. 3836–3847 (2023)

Advancing Handwritten Text Detection by Synthetic Text

Markus Muth, Marco Peer, Florian Kleber(✉), and Robert Sablatnig

Computer Vision Lab, Institute of Visual Computing and Human-Centered Technology, TU Wien, 1040 Wien, Austria
{mpeer,kleber,sab}@cvl.tuwien.ac.at

Abstract. In this paper, the usability of synthetic handwritten text to improve machine learning models is examined for the domain of handwritten text detection. We generate synthetic handwritten text by using an existing model based on a style conditioned GAN, and add those texts to scanned documents to mimic handwritten annotations. Object detection models (YOLOv5 and YOLOv8) are trained using synthetic data as a baseline to distinguish handwritten text from remaining content. We study different granularity labels (word-, line- and paragraph-level) and model sizes in our evaluation and show that applying those models to real data results in a mAP@50 of 0.88 and a pixel-level F1@50 of 0.96 for the CVL dataset, and a mAP@50 of 0.72 and F1@50 of 0.89 for SCAN, a custom dataset, created by adding real handwritten annotations to a scientific paper.

Keywords: Handwritten Text Detection · Synthetic Data · Deep Learning

1 Introduction

Handwritten Text (HWT) holds relevance across multiple domains, including gift cards, personal letters, and study materials. Its applications span from verifying signatures to conducting searches within handwritten documents [14,19]. While the main task for HWT is usually Handwritten Text Recognition (HTR), further tasks of HWT include e.g. writer identification, where factors like choice of pens or external distractions during writing are posing challenges [5,17].

However, for tasks mentioned above, preprocessing steps, e.g. layout analysis, including text detection, are necessary, to obtain HWT from the available documents. Traditionally, transcription approaches also rely on post-processing steps to extract segmented objects, such as lines or words, which significantly improve recognition accuracy [3]. While current HTR approaches massively rely on plenty of data, e.g. Fogel et al.'s ScrabbleGAN [6] uses a dataset of 100k images for word recognition, related work in the field of segmentation of HWT and printed text is sparse, for example Gholamian et al. [7] focus only on segmentation of signatures in machine printed documents.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 122–136, 2025.
https://doi.org/10.1007/978-3-031-78495-8_8

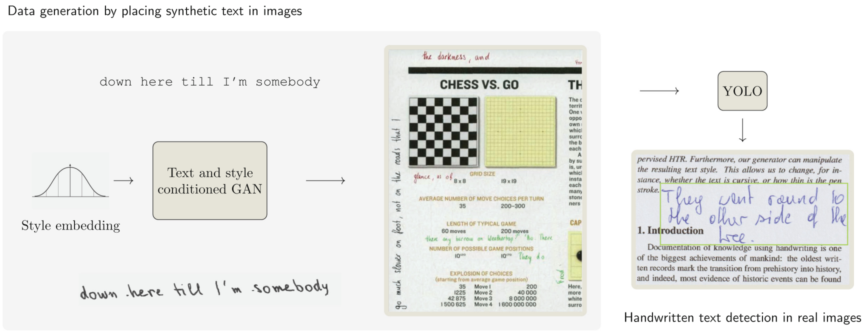

Data generation by placing synthetic text in images
down here till I'm somebody
Style embedding
Text and style conditioned GAN
down here till I'm somebody
CHESS VS. GO
YOLO
They went round to the other side of the tree.
1. Introduction
Handwritten text detection in real images

Fig. 1. Overview of our approach. First, we use the PRImA-LAD dataset and augment its images by our synthetic text to train YOLO. Our evaluation is then performed on real datasets, such as CVL or our own dataset, SCAN.

Therefore, in this paper, we want to close that gap in research and investigate the general application of synthetic data for the training of object detection models, in our case YOLO, for the detection of HWT in mixed-text scenarios - document with printed text and HWT, but might even include graphics, tables or other types of elements. Our approach does not rely on specific models or type of documents, and we use two different datasets for evaluation: CVL [13], a dataset with reduced variety in style, and SCAN, our own dataset, consisting of ten pages of the ScrabbleGAN paper printed with manually added HWT. An overview of our approach is shown in Fig. 1. For generation of synthetic text, we use the PRImA-LAD dataset as base images and train YOLO models, followed by evaluation on our two benchmark datasets.

Our results show that first of all, by only training on synthetic data, a mAP@50 of 0.88 on the CVL dataset is achieved. For the more complex SCAN dataset, we are able to outperform training on real data by using our data generation pipeline. Secondly, we study the influence of the granularity label (word, line or paragraph), and observe that line-level training works best for detecting HWT. Finally, our evaluation also includes using different model sizes, for which we do not find any significant influences, indicating that the data is the key factor for successful HWT detection.

In summary, our contributions are:

- we present a data synthesis pipeline for training HWT detection models,
- we provide a thorough evaluation of our approach on two datasets, CVL and our own created dataset called SCAN,
- and we show that with synthetic data, we achieve competitive results, reducing the need for manually annotated data.

The remainder of our paper is structured as follows: Sect. 2 describes related work in the domain of HWT generation/detection and synthetic HWT. In Sect. 3, our methodology is presented. The evaluation protocol is given in Sect. 4, and we provide results in Sect. 5. We conclude our paper in Sect. 6.

2 Related Work

This section gives a brief overview of HWT generation and HTD.

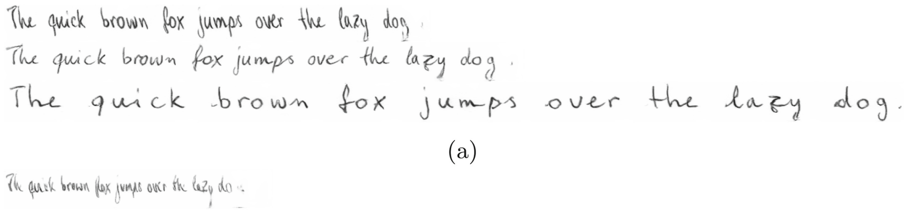
Handwritten Text Generation. Graves [8] combined LSTM and Gaussian mixture models to generate online HWT, predicting coordinates and stroke end indicators. Kang et al. [12] utilized GANs with style and textual feature encoders, optimizing for discriminative loss, style variety, and content accuracy. Fogel et al. [6] overcame string length limitations with ScrabbleGAN, generating character-conditioned patches. Davis et al. [4] employed a space predictor network for varying character widths, while Bhunia et al. [2] introduced a cycle loss for fine-grained style learning.

These models were trained on the IAM Handwriting Database [16], with Fogel et al. achieving a Fréchet Inception Distance (FID) of 23.78 and a Geometry Score (GS) of 0.00076, Davis et al. a FID of 20.65, and Bhunia et al. a FID of 19.40. Davis et al.'s model was selected for its ability to handle varying content lengths, with a human evaluation indicating 31.9% correct identification of synthetic images.

Handwritten Text Detection. Jo et al. [9] employed a CNN for pixel-level HWT segmentation on synthetic data, achieving a 92.5% accuracy. YOLOv5 and YOLOv8 object detection models are adopted as a baseline in this research, attaining mAP@50–95 of 34.3% and 37.3% on COCO data, respectively. Their adaptability and state-of-the-art performance make them suitable for Handwritten Text Detection (HTD).

Using Synthetic Handwritten Text. Jo et al. [9] augmented base images with random synthetic HWT for segmentation, resulting in annotations deemed less realistic. Fogel et al. [6] demonstrated the efficacy of their GAN-based image generation model by using synthetic images to improve HTR on the IAM Database. The augmentation with 100,000 synthetic images reduced the word error rate from 25.10% to 23.61%. This approach proves valuable for enhancing model performance on real datasets.

Figure 2 displays synthetic HWT images generated by the model developed by Davis et al. [4], showcasing the text "The quick brown fox jumps over the lazy dog." in four different styles. The visual representation highlights variations in character appearance, slant, and width, underscoring the model's versatility in generating diverse synthetic handwritten text.


The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.
The quick brown fox jumps over the lazy dog.

(a)

(b)

Fig. 2. Synthetic HWT images generated by the model developed by Davis et al. [4] with four different styles. Notable are varying appearances of the text in (a) (e.g., the letter "T" is written differently, the character slants or character widths are different). In (b) an image with a quite narrow font is shown – the smaller the character widths, the less legible the text tends to become.

3 Methodology

For the generation of synthetic handwritten text the method of Davis et al. [4], as described in Sect. 2, is used. The main reason is its ability to mimic varying pen pressures, represented by varying color intensities of the generated text. In the following, we highlight each step of the data generation pipeline, ending with describing the final synthetic datasets used for training an object detection model.

3.1 Synthetic Text Generation

The GAN proposed by Davis et al. [4] requires two inputs for generating images of handwritten text: the input text and a style vector. The character set of the input string must not deviate from the trained model. In this work, we use a pre-trained model on the IAM dataset [16] (all alphanumerical characters from the English alphabet (a–z, A–Z, 0–9), the characters `!"#&'()*+,-./:;?`, and the space character). The style vector $\mathbf{v} \sim \mathcal{N}(\mathbf{0}_{128}, \mathbf{1}_{128})$ uses a multivariate normal distributed latent space with 128 dimensions to represent the style of the handwritten text to mimic. To mimic different styles of HWT, multiple style vectors are drawn from a random distribution, which is the multivariate standard normal distribution for most datasets. However, although the generative model is trained to have a normally distributed latent space, using vectors with extreme values still provides meaningful output. Hence, style vectors following a uniform distribution over the interval $[-4, 4]$ are also used.

As shown in Fig. 2, the model from Davis et al. [4] generates images that already mimic different HWT font styles. However, all images mimic roughly the same stroke width, and the model can only generate grayscale images. Hence, the stroke width and stroke color are additionally modified to increase the diversity of styles image dilation is applied to increase the stroke width. Additionally, the foreground of the generated images is computed and applied as fully transparent alpha channel on randomly colored images to mimic different colors.

As text input for the generative model, random strings with the same character set and varying length (uniformly distributed with dataset dependent boundaries, e.g. between 1 and 90 characters) are extracted from the following text corpora:

LOTR *The Fellowship Of The Ring* novel by J. R. R. Tolkien, published in 1954.

LOB The *Lancaster-Oslo/Bergen Corpus of British English* [20], which is a collection of various British texts published in 1961. The IAM dataset [16] is based on this text collection.

GUT *Alice's Adventures in Wonderland* by Lewis Carroll, published in 1865, and *The Tragedie of Hamlet* by William Shakespeare, published in 1599.

3.2 Methodology to Generate Synthetic Datasets

The synthetic datasets for HTD are generated by adding synthetic images of HWT to "base images", which are scans of newspapers, magazines, or other documents. The HWT images are placed to mimic actual annotations, i.e., on areas where they do not overlap with any content (text, tables, images, separators, etc.) of the base images. A heuristic algorithm is applied for this purpose as follows:

1. Image selection: A base image is randomly selected from the set of all possible base images.
2. Background area selection: A rectangular area within the base image, which only contains background, is randomly selected.
3. HWT paragraph creation: Sythetically generated HWT lines are vertically stacked to form a paragraph that fits into the selected background area. Before stacking, the text is augmented regarding stroke color and stroke width, or image rotation.
4. A paragraph is placed on a random location within the selected background area. Properties of the paragraph:
   - Random number of lines up to 10, same style vector/paragraph, randomly rotated by ±1°[1]
   - Rotation by $\alpha = (a + b)^\circ$, where $a \in \{0, 90, 180, 270\}$, and $b \in [-3, 3]$
   - Random dilation with quadratic kernels k of shape 1, 2, 3, 5 or 10.
   - Inking: randomly fill with selected colors from a certain color shade (i.e. various shades of red, blue, green, or dark gray) based on the foreground mask. One paragraph has the same color.
   - Resize: Mimic varying font sizes by resizing lines from 40–200 pixels (equals font size of 12–50 pt at 300 dpi).

[1] The value of ±1° is empirically defined to keep a balance between clearly visible line rotations and extreme line spacings for paragraphs with long lines.

5. Repeat Steps 2.–4. until a randomly chosen upper limit of paragraphs has been added or no suitable background area is found[2].
6. Apply data-set dependent post-processing on the image: This is either a color-scale conversion (from RGB to gray-scale or binarized black-white images using Otsu), or creation of cutouts of size 640 × 640 pixels to meet the requirements on the size of input images for YOLOv5, or both, or none of them.
7. Start again at Step 1. Until the desired number of synthetically annotated documents has been created.

The base images are taken from the PRImA-LAD dataset [1], a layout analysis dataset containing scans of over 400 pages from magazines or technical articles. Each scan is accompanied by a detailed description of the layout of the page (e.g. information about regions containing charts, images, noise, separators, text, or other content), allowing to distinguish foreground and background areas. 382 images with a width from 2080–4808 pixels, a height from 2858–3533 pixels, and a resolution of 300 dpi remain after sorting out the images with an incomplete layout description or containing areas with HWT. Those 382 scans are the base images for the synthetic HTD datasets; one such base image is possibly used multiple times to generate synthetic images with more than 382 pages, but the synthetic annotations are added with a different random seed.

Label Granularity: The synthetic HTD dataset labels are the bounding boxes of areas containing HWT for different label categories: entire paragraphs (PAR), lines (LINE), or single words (WORD).

Since real annotations are always found in empty spaces on a page, we assume that the artificially generated data set matches real data.

3.3 Parameters of Generated Synthetic Datasets

Different datasets are generated to control for various aspects of their properties, especially color scales, HWT stroke width, and label granularity. The following list shows the synthetic HWT datasets with the according number of training, validation, and test images (train/val/test):

- COLSCALES-RGB-CUT-PAR (32,814/7,380/6,335)
- COLSCALES-GRAY-CUT-PAR (32,814/7,380/6,335)
- COLSCALES-BW-CUT-PAR (32,814/7,380/6,335)
- COLSCALES-STROKE-BW-CUT-PAR (32,946/7,353/6,398)
- FULLSIZE-STROKE-BW-PAR (1,412/312/276)
- GRANULARITY-STROKE-BW-PAR (1,401/309/275)
- GRANULARITY-STROKE-BW-LINE (1,401/309/275)
- GRANULARITY-STROKE-BW-WORD (1,401/309/275)
- CWT-STROKE-BW-PAR (2,401/309/275)

[2] Each document has a randomly chosen number of 1 .. 12 paragraphs (those boundaries are empirically defined). The algorithm stops after 1000 iterations (also empirically defined), hence less paragraphs than this randomly chosen upper limit are possible as well.

- CWT-STROKE-BW-LINE (2,401/309/275)
- CWT-STROKE-BW-WORD (2,401/309/275)

The *COLSCALES-** datasets are all based on 2,000 synthetically annotated documents, hence the base images from the PRImA-LAD dataset occur on average more than five times. The actual training data is based on cutouts (CUT) of the images with size 640×640 pixels and a stride of 160 pixels. The images are downscaled by 50% before creating the cutouts to reduce the total number of images. The training, validation and test sets contain 32,814, 7,380 and 6,335 images, respectively. Color augmentation is applied to the HWT before adding it to the base images. *-RGB-* contains RGB images, *-GRAY-* contains only grayscale images, and *-BW-* only binarized data. All labels are on paragraph (PAR) level.

Further analyzing the synthetic images reveals that the stroke width of HWT text is generally relatively thin compared to real-world images, e.g., from the CVL dataset [13]. Hence, stroke width augmentation is introduced for the dataset *-STROKE-*, which is equal to *COLSCALES-RGB-CUT-PAR* with all other parameters.

Dataset *FULLSIZE-STROKE-BW-PAR* contains the full size images (used for YOLOv8), a stroke width augmentation, binarized images and labels on paragraph (PAR) level.

Generating synthetically annotated documents allows full control over the granularity of the ground truth, i.e., about the position of paragraphs, lines, or single words. The datasets *GRANULARITY-STROKE-PAR*, *GRANULARITY-STROKE-LINE*, and *GRANULARITY-STROKE-WORD* all have equal configuration, but labels on paragraph (PAR), line (LINE), and word (WORD) level, respectively.

The base images from the PRImA-LAD collection usually have the content centered within the image - often using a two-column layout. Hence, the generated images have most of the HWT content towards the border of the documents. To generate additional images with a more diverse distribution of HWT content, different kinds of synthetic documents (CWT-, Computer Written Text) are added to the dataset: Paragraphs based on the *LOTR* corpus using computer fonts, and synthetic HWT text paragraphs, are randomly placed on empty images, which have a width and height randomly chosen and limited by the minimum and maximum width and height of the images from the PRImA-LAD dataset. The datasets *CWT-STROKE-BW-PAR*, *CWT-STROKE-BW-LINE*, and *CWT-STROKE-BW-WORD* are equal to *GRANULARITY-STROKE-BW-**, except their training data is extended with 1000 of such synthetic images.

4 Evaluation

HTD is approached by training object detection models on the synthetic data and assessing their performance using evaluation datasets with real handwritten text. The deep learning models YOLOv5 [10] and YOLOv8 [11] by Ultralytics are used for this purpose. All models are trained on computers of the Vienna

Scientific Cluster, more precisely the VSC-5, which offers two different GPU-enabled nodes containing either 2x NVIDIA A40 or 2x NVIDIA A10 GPUs.

4.1 Evaluation Datasets

To evaluate the performance, the following two document image analysis datasets are used: the first one is the CVL-Database [13], which contains in total 1604 scanned pages with CWT and HWT. The images contain an area of CWT on the top, which are transcribed by, in total, 310 participants beneath this area. Seven different texts are transcribed, of which one was in German, the others in English. Due to the limitation of the character set of the trained HWT model, texts with German umlauts have been excluded. The dataset is split into 189 training images and 1415 test images. All images are accompanied by a ground truth containing the bounding boxes and content of the entire HWT area, single lines, and individual words. The scans are available in RGB, but most HWT was written with a blue pen. Further summary of the properties of the CVL dataset are shown in Table 1.

The second dataset, called SCAN dataset, contains a scan of the paper "*ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation*" [6], which was manually annotated by two persons with random text lines from [21]. Ten images are available in total. The annotations are made with different pens and pen colors. Each image contains various annotations, most of which contain multiple lines. Labels are created manually for paragraph-, line- and word-level by one person using the web-based image annotation software DataTorch[3]. Figure 3 shows an example image of this dataset. Further details can be found in Table 1.

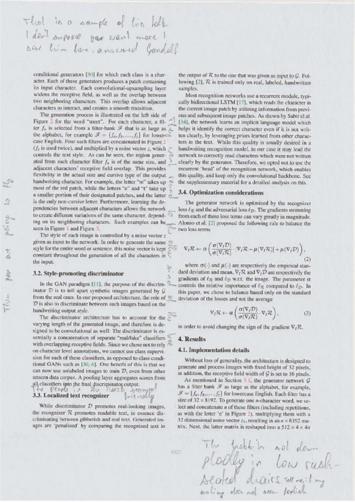
4.2 Experimental Setup

For the evaluation the model architectures YOLOv5n (5n) [10], YOLOv8n (8n - nano), and YOLOv8m (8 m - medium) [11] are used. If not stated otherwise, models are pre-trained on the COCO dataset [15]. All models are trained with the implementations' default optimizer, stochastic gradient descent with Nesterov momentum, with learning rate $\gamma = 0.1$ and momentum $\mu = 0.937$. Early stopping is enabled, with the default patience of 100 iterations for YOLOv5 models and 50 iterations for YOLOv8 models. The maximum epochs per training run is limited to 1000. The models are trained with the implementations' default hyperparameters.

4.3 Metrics

As metric, the Mean Average Precision (mAP), which is the mean of all Average Precisions (AP) for all classes is used. Since the HTD tasks have one class only, this is equal to the average precision of this class. The mAP is often interpreted

[3] https://datatorch.io, last accessed on 2023-07-16.



That is a sample of his talk. I don't suppose you want more I saw him here, answered Gandalf

conditional generators [30] for which each class is a character. Each of these generators produces a patch containing its input character. Each convolutional-upsampling layer widens the receptive field, as well as the overlap between two neighboring characters. This overlap allows adjacent characters to interact, and creates a smooth transition.

The generation process is illustrated on the left side of Figure 2 for the word "meet". For each character, a filter f_* is selected from a filter-bank $\mathscr{F}$ that is as large as the alphabet, for example $\mathscr{F} = \{f_a, f_b, \dots, f_z\}$ for lowercase English. Four such filters are concatenated in Figure 2 (f_e is used twice), and multiplied by a noise vector z, which controls the text style. As can be seen, the region generated from each character filter f_* is of the same size, and adjacent characters' receptive field overlap. This provides flexibility in the actual size and cursive type of the output handwriting character. For example, the letter "m" takes up most of the red patch, while the letters "e" and "t" take up a smaller portion of their designated patches, and the latter is the only non-cursive letter. Furthermore, learning the dependencies between adjacent characters allows the network to create different variations of the same character, depending on its neighboring characters. Such examples can be seen in Figure 1 and Figure 3.

The style of each image is controlled by a noise vector z given as input to the network. In order to generate the same style for the entire word or sentence, this noise vector is kept constant throughout the generation of all the characters in the input.

3.2. Style-promoting discriminator

In the GAN paradigm [11], the purpose of the discriminator $\mathcal{D}$ is to tell apart synthetic images generated by $\mathcal{G}$ from the real ones. In our proposed architecture, the role of $\mathcal{D}$ is also to discriminate between such images based on the handwriting output *style*.

The discriminator architecture has to account for the varying length of the generated image, and therefore is designed to be convolutional as well: The discriminator is essentially a concatenation of separate "real/fake" classifiers with overlapping receptive fields. Since we chose not to rely on character level annotations, we cannot use class supervision for each of these classifiers, as opposed to class conditional GANs such as [30, 6]. One benefit of this is that we can now use unlabeled images to train $\mathcal{D}$, even from other unseen data corpus. A pooling layer aggregates scores from all classifiers into the final discriminator output.

The people in the marsh seemed friendly

3.3. Localized text recognizer

While discriminator $\mathcal{D}$ promotes real-looking images, the recognizer $\mathcal{R}$ promotes readable text, in essence discriminating between gibberish and real text. Generated images are 'penalized' by comparing the recognized text in

Then you are going to fly

They went round to the other side of the tree

the output of $\mathcal{R}$ to the one that was given as input to $\mathcal{G}$. Following [2], $\mathcal{R}$ is trained only on real, labeled, handwritten samples.

Most recognition networks use a recurrent module, typically bidirectional LSTM [17], which reads the character in the current image patch by utilizing information from previous and subsequent image patches. As shown by Sabir el al. [34], the network learns an implicit language model which helps it identify the correct character even if it is not written clearly, by leveraging priors learned from other characters in the text. While this quality is usually desired in a handwriting recognition model, in our case it may lead the network to correctly read characters which were not written clearly by the generator. Therefore, we opted not to use the recurrent 'head' of the recognition network, which enables this quality, and keep only the convolutional backbone. See the supplementary material for a detailed analysis on this.

3.4. Optimization considerations

The generator network is optimized by the recognizer loss $\ell_{\mathcal{R}}$ and the adversarial loss $\ell_{\mathcal{D}}$. The gradients stemming from each of these loss terms can vary greatly in magnitude. Alonso et al. [2] proposed the following rule to balance the two loss terms

$$\nabla_I \mathcal{R} \leftarrow \alpha \left(\frac{\sigma(\nabla_I \mathcal{D})}{\sigma(\nabla_I \mathcal{R})} \cdot [\nabla_I \mathcal{R} - \mu(\nabla_I \mathcal{R})] + \mu(\nabla_I \mathcal{D}) \right), \tag{2}$$

where $\sigma(\cdot)$ and $\mu(\cdot)$ are respectively the empirical standard deviation and mean, $\nabla_I \mathcal{R}$ and $\nabla_I \mathcal{D}$ are respectively the gradients of $\ell_{\mathcal{R}}$ and $\ell_{\mathcal{D}}$ w.r.t. the image. The parameter α controls the relative importance of $\ell_{\mathcal{R}}$ compared to $\ell_{\mathcal{D}}$. In this paper, we chose to balance based only on the standard deviation of the losses and not the average

$$\nabla_I \mathcal{R} \leftarrow \alpha \left(\frac{\sigma(\nabla_I \mathcal{D})}{\sigma(\nabla_I \mathcal{R})} \cdot \nabla_I \mathcal{R} \right), \tag{3}$$

in order to avoid changing the sign of the gradient $\nabla_I \mathcal{R}$.

4. Results

4.1. Implementation details

Without loss of generality, the architecture is designed to generate and process images with fixed height of 32 pixels, in addition, the receptive field width of $\mathcal{G}$ is set to 16 pixels.

As mentioned in Section 3.1, the generator network $\mathcal{G}$ has a filter bank $\mathscr{F}$ as large as the alphabet, for example, $\mathscr{F} = \{f_a, f_b, \dots, f_z\}$ for lowercase English. Each filter has a size of 32×8192. To generate one n-character word, we select and concatenate n of these filters (including repetitions, as with the letter 'e' in Figure 2), multiplying them with a 32 dimensional noise vector z_1, resulting in an $n \times 8192$ matrix. Next, the latter matrix is reshaped into a $512 \times 4 \times 4n$

4327

The hobbits sat down gladly in low rush-seated chairs. Tell me, if my asking does not seem foolish

Fig. 3. A sample image from the *SCAN* dataset with manual annotations from [21].

Table 1. Summary of the evaluation datasets for HTD. If multiple values are listed in a table cell, then they refer to the paragraph-, line- and word-level datasets, respectively.

Property	CVL	SCAN
Image width	2,480–2,663 px	2,473 px
Image height	3,507–3,634 px	3,495 px
#images	1,411	10
#objects	1,409/11,849/ 88,825	65/228/877
#objects/image (average)	1/8/63	6/23/88

at a certain threshold of the Intersection Over Union (IoU), which indicates how much a predicted bounding box overlaps with a bounding box from the ground truth. The typical threshold of 0.5 is used in this work (mAP@50), meaning that the IoU must be at least 50%.

The remaining metrics reported are based on pixel-level comparisons of the predictions vs. ground truth using precision (P), recall (R), and F1 score (F1). Only predictions with a confidence level of at least 0.5 are considered. Therefore, the metrics P@50, R@50, and F1@50 are the averages of the pixel-level precision, recall, and F1 score of all images considering predictions with a confidence of at least 0.5. The results are reported using the test set of the synthetic data, as well as the *CVL* and *SCAN* datasets.

5 Results

In this section, we report the main findings of our studies. Additionally, we evaluate the influence of different setting on our approach, such as color scale, data augmentation and label granularity.

5.1 Color Scale and Data Augmentation

Table 2a lists results for models trained on two different kinds of datasets: *COLSCALES-*-CUT-PAR* with images in different color scales (RGB, grayscale and binarized images), and *COLSCALES-STROKE-BW-CUT-PAR* with binarized images and stroke width augmentation. Images with RGB color scale provide the best results considering the synthetic data only: the mAP@50 is 0.86, the F1@50 is 0.89. Introducing stroke width augmentation has no further effect for the synthetic data but increases the results on the CVL and SCAN dataset.

Using grayscale or binarized data yields strictly better results when evaluating on real data, likely due to less details the model has to filter out. Grayscale vs. binarized images do not have a significant difference. However, using binarized images provides the best results for the *SCAN* datasets although stroke width augmentation has different effects on F1@50 and mAP@50, but image binarization together with stroke width augmentation provides the best results for the *CVL* data. Hence, all following experiements are done one binarized images with stroke width augmentation.

Table 2. Description and metrics for HTD models trained on (a) different datasets (model architecture 5n, image size 640, and dataset (COLSCALES-*-CUT-PAR) and (b) using different model architectures and image sizes on dataset FULLSIZE-STROKE-BW-PAR.

(a) Dataset Evaluation						(b) Model Architecture Evaluation					
Test Set	Model	P@50	R@50	F1@50	mAP@50	Test Set	Model	P@50	R@50	F1@50	mAP@50
SYN	RGB	**0.83**	**0.97**	**0.89**	**0.86**	SYN	5n, 1280	0.95	0.93	0.94	0.95
	GRAY	0.82	**0.97**	**0.89**	0.84		8n, 640	0.95	0.94	0.95	0.87
	BW	**0.83**	0.96	**0.89**	0.82		8n, 1280	0.97	0.95	0.96	0.94
	STROKE-BW	0.82	0.96	**0.89**	0.82		8m, 1280	**0.98**	**0.97**	**0.97**	**0.96**
CVL	RGB	**0.91**	0.85	0.88	0.27	CVL	5n, 1280	0.94	0.14	0.24	0.02
	GRAY	0.90	0.88	0.89	0.35		8n, 640	**0.96**	**0.84**	**0.90**	**0.78**
	BW	**0.91**	0.80	0.85	0.42		8n, 1280	0.95	0.76	0.85	0.48
	STROKE-BW	0.89	**0.90**	**0.90**	**0.47**		8m, 1280	0.95	0.77	0.85	0.38
SCAN	RGB	0.60	0.35	0.44	0.17	SCAN	5n, 1280	0.84	**0.80**	**0.82**	0.51
	GRAY	0.76	**0.82**	0.79	0.55		8n, 640,	0.87	0.67	0.75	**0.53**
	BW	**0.83**	0.80	**0.81**	0.52		8n, 1280	**0.89**	0.76	**0.82**	0.44
	STROKE-BW	0.65	0.74	0.69	**0.56**		8m, 1280	**0.89**	0.69	0.78	0.41

5.2 Model Architecture and Image Size

It can be seen that YOLOv8 outperforms YOLOv5, as shown in Table 2b. The best results on synthetic data are achieved using a YOLOv8m model with training data sized 1280 × 1280 pixels; the mAP@50 is 0.96, the F1@50 is 0.97.

The best model regarding the *CVL* data is a YOLOv8n model trained on cutouts of 640 × 640 pixels: the mAP@50 is 0.78, the F1@50 is 0.90. However, the mAP@50 for the other models is significantly lower, and has with 0.02 its minimum; but the best performing model on the synthetic data (YOLOv8m with image sizes pf 1280 × 1280 pixels) still yields a F1@50 of 0.85, whereas the mAP@50 is 0.38. The precision of 0.95 indicates that HWT was correctly identified in the sense that if the model predicted an HWT bounding box, it contained HWT to a large extent. But since the paragraphs for the *CVL* data are relatively large compared to the synthetic images (due to the layout of the CVL dataset), the predicted bounding boxes also need to be larger, otherwise the predictions would be too small. This would result in a low recall, a low F1, a low IoU, many false positives, and finally a low mAP@50. Using cutouts mitigates this problem, as the distribution of bounding box sizes is more equal for synthetic and *CVL* data.

Table 3. Description and metrics for HTD models trained for (a) different label granularities (dataset GRANULARITY-STROKE-BW-; model architecture 8n, image size 1280) and (b) with datasets containing synthetic CWT (trained on CWT-STROKE-BW-; image size 1280).

(a) Label Granularity Evaluation

Test Set	Model	P@50	R@50	F1@50	mAP@50
SYN	PAR	**0.97**	0.96	**0.97**	0.94
	LINE	**0.97**	**0.97**	**0.97**	**0.97**
	WORD	0.96	0.96	0.96	0.89
CVL	PAR	**0.98**	0.76	0.85	0.13
	LINE	0.94	**0.92**	**0.93**	**0.79**
	WORD	0.88	0.86	0.87	0.76
SCAN	PAR	**0.96**	0.74	0.84	0.30
	LINE	**0.96**	**0.81**	**0.88**	0.56
	WORD	0.95	0.69	0.80	**0.67**

(b) Synthetic CWT Evaluation

Test Set	Model	P@50	R@50	F1@50	mAP@50
SYN	8n, PAR	0.97	0.97	0.97	0.94
	8n, LINE	**0.98**	0.97	0.97	**0.97**
	8n, WORD	0.95	0.97	0.96	0.90
	8m, PAR	**0.98**	**0.98**	**0.98**	0.96
	8m, LINE	**0.98**	**0.98**	**0.98**	**0.97**
CVL	8n, PAR	0.98	**0.93**	0.95	0.84
	8n, LINE	0.94	**0.93**	0.93	0.82
	8n, WORD	0.88	0.88	0.88	0.78
	8m, PAR	**0.99**	**0.93**	**0.96**	**0.88**
	8m, LINE	0.95	**0.93**	0.94	0.84
SCAN	8n, PAR	**0.98**	0.80	0.88	0.37
	8n, LINE	0.96	**0.83**	**0.89**	0.62
	8n, WORD	0.95	0.77	0.85	**0.72**
	8m, PAR	0.96	0.75	0.84	0.48
	8m, LINE	0.97	**0.83**	**0.89**	0.59

5.3 Label Granularity

Table 3a summarizes the results for models trained with equal data with respect to the granularity, but different labels.

Notably is, again, that using labels on paragraph level yields the lowest mAP@50 (0.13 for the *CVL* data, 0.30 for the *SCAN* data), while the precision is at 0.98 for the *CVL* data and 0.96 for the *SCAN* data. Reducing the label granularity increases both the F1@50 and the mAP@50 for both real datasets. Using labels on line-level provides the best results for the synthetic data (mAP@50 and F1@50 are both 0.97) and for the *CVL* data (mAP@50 is 0.79, F1@50 is 0.93). Although the F1@50 using line-level labels is with 0.88 the highest for the *SCAN* data, the highest mAP@50 of 0.67 is achieved using labels on word-level.

5.4 Diverse Synthetic Layout Datasets

Another approach to align the bounding box distributions is to add bigger HWT paragraphs, as done with the *CWT-STROKE-BW-** datasets, which are equal to the *GRANULARITY-STROKE-BW-** datasets but with additional 1,000 fully synthetic images. The results for models trained on this data are listed in Table 3b.

Notably is the mAP@50 for the *CVL* and *SCAN* data when trained on the YOLOv8 model compared with the results of the same model architecture shown in Table 3a. Adding fully synthetic data with bigger paragraphs increased the mAP@50 from 0.13 to 0.84 for the *CVL* data, and from 0.30 to 0.37 for the *SCAN* data.

The main motivation for using this dataset is to allow a more diverse placement of HWT. Magazines or papers tend to have most of the content centered in the page, with only limited space between paragraphs or columns. Hence, most of the HWT must be placed towards the borders of the image; using completely synthetic data, where CWT and HWT are randomly placed, allows to break up this structure.

However, this approach yields little improvement using labels on line and word level compared to labels on paragraph level. For the synthetic data, the mAP@50 increased only for labels on word-level, from 0.89 (in Table 3a) to 0.90. Using the *CVL* dataset and labels on line-level, the mAP@50 increased from 0.79 to 0.82, the F1@50 is unchanged; using labels on word-level increases the mAP@50 from 0.76 to 0.78, the F1@50 from 0.87 to 0.88. A stronger effect is observed on the *SCAN* data: on line level, the mAP@50 is increased from 0.56 to 0.62, the F1@50 from 0.88 to 0.89; on word level, the mAP@50 is increased from 0.67 to 0.72, the F1@50 from 0.80 to 0.84.

Using YOLOv8m models instead of YOLOv8n further increases the performance when evaluated on synthetic or *CVL* data. The best results for those datasets are achieved with labels on line level.

5.5 HTD Baseline

Table 4 shows the results of HTD if trained on the CVL train set (real HWT). The results show that almost perfect results are achieved on the CVL test set (due to its simple layout). However, the CVL trained HTD drops to a mAP@50 of almost 0 for the SCAN dataset which represents realistic annotated documents.

Table 4. Description and metrics for HTD models trained for different label granularities on the CVL train set (real data, model architecture 8n, image size 1280).

Test Set	Model	P@50	R@50	F1@50	mAP@50
CVL	PAR	1	1	0.99	0.99
	LINE	0.97	0.988	0.97	0.987
	WORD	0.90	0.869	0.88	0.90
SCAN	PAR	0.01	0.17	0.02	0.01
	LINE	0.25	0.04	0.07	0.04
	WORD	0.41	0.24	0.30	0.24

6 Conclusion

The potential of using synthetic datasets is examined for the domain of handwritten text detection. Synthetic HWT is added to scans of documents to mimic human annotations. Object detection models (YOLOv5 and YOLOv8) are trained on this data to distinguish HWT from remaining content in the documents. The suitability of using synthetic data is evaluated by assessing the performance of those models on two datasets containing real HWT: the *CVL* dataset [13], and a new *SCAN* dataset, where handwritten text is manually added to a scientific paper.

The models are trained with different granularity of the labels: paragraphs, lines, and single words. The best models on the *CVL* dataset achieved a mAP@50 of 0.88 and a F1@50 of 0.96 on paragraph level, a mAP@50 of 0.84 and F1@50 of 0.94 on line level, and a mAP@50 of 0.78 and F1@50 of 0.88 on word level; For the *SCAN* dataset, a mAP@50 of 0.48 and a F1@50 of 0.84 are achieved on paragraph level, a mAP@50 of 0.62 and F1@50 of 0.89 on line level, and a mAP@50 of 0.72 and F1@50 of 0.85 on word level. The best result is usually achieved on the synthetic data, as this corresponds to the training set. However, it is shown that comparable results can be achieved on real data. Detailed discussions are presented in Muth [18].

Summarized, synthetic HWT allows the generation of images tailored to the target data based on text size, stroke width, layout, color, and text granularity (line, word and paragraph based annotation). Thus, using synthetic data allows full control over the properties of the training data and can be used in the domain of HTD to achieve comparable results to real data.

References

1. Apostolos, A., Bridson, D., Papadopoulos, C., Pletschacher, S.: A realistic dataset for performance evaluation of document layout analysis. In: Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR2009), pp. 296–300 (2009)

2. Bhunia, A.K., Khan, S., Cholakkal, H., Anwer, R.M., Khan, F.S., Shah, M.: Handwriting Transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1086–1094 (2021)
3. Carbonell, M., Mas, J., Villegas, M., Fornés, A., Lladós, J.: End-to-End Handwritten Text Detection and Transcription in Full Pages. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), vol. 5, pp. 29–34 (2019)
4. Davis, B.L., Morse, B.S., Price, B.L., Tensmeyer, C., Wigington, C., Jain, R.: Text and style conditioned GAN for generation of offline handwriting lines. In: 31st British Machine Vision Conference 2020, BMVC 2020, Virtual Event, UK (2020)
5. Fiel, S., Sablatnig, R.: Writer identification and retrieval using a convolutional neural network. In: CAIP 2015, Part II, pp. 26–37 (2015)
6. Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., Litman, R.: ScrabbleGAN: semi-supervised varying length handwritten text generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
7. Gholamian, S., Vahdat, A.: Handwritten and printed text segmentation: a signature case study. In: IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, 1–6 October 2023, pp. 582–592 (2023)
8. Graves, A.: Generating sequences with recurrent neural networks. CoRR abs/1308.0850 (2013)
9. Jo, J., Koo, H.I., Soh, J.W., Cho, N.I.: Handwritten text segmentation via end-to-end learning of convolutional neural networks. Multimed. Tools Appl. **79**(43–44), 32137–32150 (2020)
10. Jocher, G., et al.: Ultralytics/YOLOv5: v7.0 - YOLOv5 SOTA realtime instance segmentation (2022). https://zenodo.org/record/7347926. Accessed 2023-09-24
11. Jocher, G., Chaurasia, A., Qiu, J.: YOLO by Ultralytics. version 8.0.0 (2023). https://github.com/ultralytics/ultralytics. Accessed 24 Sept 2023
12. Kang, L., Riba, P., Wang, Y., Rusiñol, M., Fornés, A., Villegas, M.: GANwriting: content-conditioned generation of styled handwritten word images. In: ECCV 2020, pp. 273–289 (2020)
13. Kleber, F., Fiel, S., Diem, M., Sablatnig, R.: CVL-DataBase: an off-line database for writer retrieval, writer identification and word spotting. In: 2013 12th International Conference on Document Analysis and Recognition (ICDAR), pp. 560–564 (2013)
14. Krishnan, P., Dutta, K., Jawahar, C.: Deep feature embedding for accurate recognition and retrieval of handwritten text. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 289–294 (2016)
15. Lin, T.Y., et al.: Microsoft COCO: common objects in context. In: ECCV 2014, pp. 740–755 (2014)
16. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recogn. **5**(1), 39–46 (2002)
17. Marti, U.V., Messerli, R., Bunke, H.: Writer identification using text line based features. In: Proceedings of Sixth International Conference on Document Analysis and Recognition, pp. 101–105 (2001)
18. Muth, M.M.: Synthetic data for applications in document analysis. Diploma Thesis (2023). https://repositum.tuwien.at/handle/20.500.12708/188733. Artwork Size: 69 pages, TU Wien
19. Shen, Q., Luan, F., Yuan, S.: Multi-scale residual based Siamese neural network for writer-independent online signature verification. Appl. Intell. **52**(12), 14571–14589 (2022)

20. Stig, J., Leech, G.N., Goodluck, H.: Manual of information to accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with digital computers. Department of English, University of Oslo (1978)
21. Tolkien, J.R.R.: The fellowship of the ring. The Lord of the Rings, HarperCollins, London, England (2020)

Learning-Based Sub-image Retrieval in Historical Document Images

Joseph Assaker(✉), Stéphane Nicolas, and Laurent Heutte

Univ Rouen Normandie, LITIS UR 4108, 76000 Rouen, France
{joseph.assaker,stephane.nicolas,laurent.heutte}@univ-rouen.fr

Abstract. The goal of this paper is to propose an unsupervised learning-based framework in order to deal with any kind of one-shot object detection scenario, focusing on the tasks of sub-image retrieval and pattern spotting in historical document images. Taking in an arbitrary object/pattern query from users, the proposed framework should be able to retrieve images containing it, as well as localising each occurrence within the images. A major difficulty is the lack of any training data. Three contributions are thus presented: (1) a novel model architecture dubbed OS-DETR, capable of adapting to various tasks by simply swapping training data, (2) a completely unsupervised synthetic data generation process, easily applicable to many data-limited domains, and (3) a set of training strategies catered to boost the model's generalisation capabilities. The result is a framework that yields a strong baseline for learning-based approaches applied to sub-image retrieval and pattern spotting.

Keywords: Sub-Image Retrieval · Pattern Spotting · Image Retrieval · One-Shot Object Detection · Historical Document Images

1 Introduction

Finding occurrences of a specific object instance in a collection of images is a core task in computer vision with a plethora of applications. Either be it finding a specific logo in a collection of product images, a specific stamp on parcels, or a specific pattern in a collection of historical document images. In fact, the latter could become a primordial tool for historians that significantly upgrades their analysis capabilities when studying relationships, circulation and provenance of specific patterns. Most works approaching this problematic in the vision domain belong to the sub-field of one-shot object detection, which deals with natural world images, treating objects at a semantic level as well as benefiting from a large training corpus from the same distribution [10].

In this work we address the tasks of sub-image retrieval and pattern spotting in historical document images (Fig. 1). Whereas image retrieval is traditionally defined as the task of retrieving (from a collection of images) images similar to an input image, here we define sub-image retrieval as the task of retrieving

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 137–151, 2025.
https://doi.org/10.1007/978-3-031-78495-8_9

images that *contain* an input sub-image (usually a cropped sub-image of an object/pattern). As multiple occurrences of a given sub-image can be found within a single retrieved image, and as localization information can be utilised in a multitude of down-stream tasks (e.g., explainability, document layout analysis, etc.), pattern spotting is defined as the task of retrieving all occurrences (i.e., with localization information) of a given queried sub-image in a collection of images. The challenges in these tasks are two-fold. (1) Users (such as historians) could choose to query any kind of pattern when using the system, as it is not possible to know what could interest them during their analyses in advance. Thus, training a standard detection model specific for a predefined set of N classes is unfeasible. Also, the word *class* does not really fit in this context, as we are aiming to retrieve occurrences of the same object instance portrayed in the query image, at an almost near-duplicate level. (2) We do not have access to any labelled training data to use in our work. This fact has led all previous works [4,7,19] to propose learning-free approaches, by essentially pre-computing image features (using hand-crafted representations or pre-trained deep architectures) in an *offline phase* and measuring distances with the query features in an *online phase*.

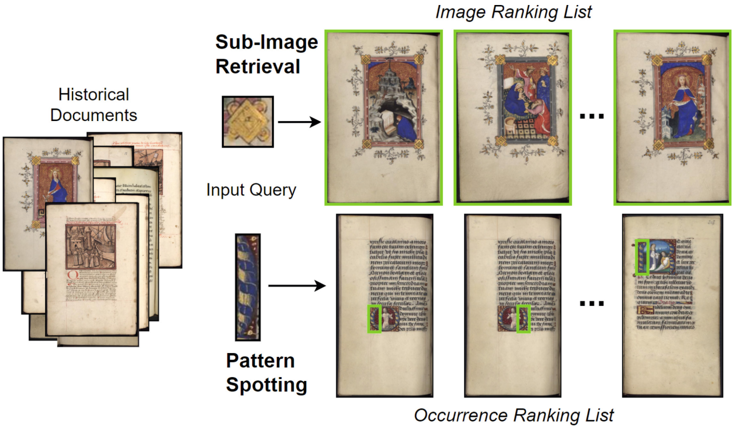


Fig. 1. Given an input query image, Sub-Image Retrieval (top) returns a ranked list of images that contain the query, and Pattern Spotting (bottom) returns a ranked list of occurrences of the query within images. All data shown are from the DocExplore dataset [6].

In this work, we aim to present a learning-based framework in order to deal with any kind of one-shot object detection scenario, focusing on the tasks of sub-image retrieval and pattern spotting in historical document images. After presenting relevant state of the art in Sect. 2, we will present our flexible model architecture in Sect. 3 whose components can be swapped easily for future improvements, a synthetic data generation process that can easily be replicated/tuned for other domains in Sect. 4, a set of training strategies that boost generalisation and are catered for the challenges faced in sub-image retrieval and pattern spotting in Sect. 5. Finally, we will see how the training objective can be easily modified and refined based only on the training data as shown in both the preliminary results and in Sect. 6, in which results on a synthetic test set and the DocExplore dataset prove the efficacy of the proposed approach.

2 State of the Art

An object detector takes as input an image, and aims to detect occurrences of predefined classes within the input image. For each class we aim to detect, an adequately large sample of labelled examples is expected to be provided to the model in the training stage. Object detectors can be categorised into 2 categories, CNN-based object detectors and Transformer-based object detectors. CNN-based object detectors, such as Faster R-CNN [15], have been dominating benchmarks for the better part of the last decade, that was until the emergence of transformer-based object detectors, and more specifically DETR [2] and its variants [5,13]. On top of its remarkable performance, DETR differentiates itself from other CNN-based approaches by the fact that it is a completely end-to-end architecture. For instance, there is no explicit region proposal or anchor-based predictions in the model, and there is not even the need for any post-processing of the output bounding boxes, due to its set-based global loss that forces unique predictions via bipartite matching.

On the other hand, one-shot object detection aims to generalise the task of object detection to any class queried at test time, and not only having the model limited to detecting a predefined set of classes seen during training. As defined in the literature [10], it is the task of taking in a pair of images, a query patch and a target image, and to detect occurrences of the query class within the target image. The term *one-shot* refers to the fact that the model has only access to a single example (a single cropped image) of the class we are aiming to detect, implying the absence of such a class during training. That being said, and even though classes reserved for testing are hidden during training, these approaches still have access to a large corpus of training data from the same domain/distribution. So learned characteristics and semantics can still be very useful when addressing the task even on unseen data.

Many one-shot object detection proposals are FRCNN-based [3,10]. Meaning they alter the FRCNN architecture such that it becomes capable of performing the one-shot task. One key limitation in these approaches however is the region proposal network, which once trained on seen classes, has difficulties in predicting reasonable regions for unseen classes, which in turn translates into a bottleneck in the whole architecture. For our work we opted to build a DETR-based architecture, being a completely end-to-end architecture making it more attractive for us to adapt for our task. In fact, a handful of works also opted to adapt DETR to one-shot [5] or few-shot [18] scenarios for object detection, however all these efforts remain focused on the natural world domain, and more specifically on the COCO dataset. To the best of our knowledge, our work is the first attempt to apply such a one-shot task to historical documents.

In this work we address the tasks of sub-image retrieval and pattern spotting in historical document images. As mentioned earlier, one of the key limitations and differentiation between the tasks we are tackling and standard one-shot object detection, is that there is no labelled data that could be utilised for training purposes. In fact, the only effort of annotation for these tasks that we are aware of in the historical document domain is the DocExplore dataset [6]. That being said, and given the relatively small scale of this dataset, it is generally reserved exclusively for testing purposes. This makes these tasks automatically harder to tackle, especially for learning-based approaches. This is why, all previous efforts on these tasks avoided learning-based approaches and instead limited themselves to only make use of pre-trained networks in order to extract image features and compute correlations between query and target images. En et al. [7] relied on hand-crafted representations in order to generically encode information from image data. Ubeda et al. [19] on the other hand utilised deep features generated by a pre-trained feature pyramid network (FPN) without any fine-tuning, and computed cross-correlation between extracted embeddings. More recently, Curi et al. [4] significantly improved performance by utilising fully convolutional networks to be able to process queries and documents of various sizes with no need for preprocessing such as resizing, further pushing the state of the art on the DocExplore dataset.

All mentioned approaches suffer from the same limitations: (i) Purely relying on the features provided by pre-trained networks, with no further processing other than correlation computation, leading to having semantics limited by what has been learned in pre-training, and no adaptation to the current domain/task. (ii) By computing cross-correlation between query and target images these approaches are effectively limited to only detecting same-sized occurrences of the query image. This is unfortunately not punished in the DocExplore testing dataset, as even though different patterns in the dataset vary in size, the occurrences of a given pattern generally have the same size. In contrast, our approach, as will be seen in the following sections, does overcome both of these limitations.

3 Proposed Architecture

In order to solve the aforementioned tasks, we propose the first learning-based approach that aims to pave the way to better and more flexible solutions down the road. We will start by presenting our simple yet efficient model architecture visualised in Fig. 2.

3.1 Model Architecture

We decided to base our work on the DETR [2] architecture given its end-to-end nature. The absence of many hand-crafted modules in this detection system especially attracted us. So our architecture will effectively modify the DETR architecture in order to adapt to one-shot scenarios. Firstly, and as it is the norm in one-shot object detection, our architecture will take 2 images as input, the target image and the query image. We will adopt the usage of a CNN backbone in conjunction with the transformer-based rest of the architecture. A plethora of backbone choices are present to us, and in this work we opt to use an ImageNet pretrained ResNet50 as our CNN backbone. This is similar to what is done in the original DETR as well as in many newer DETR versions [5,13]. Most notably, this will help both in terms of reducing the computational complexity of our architecture as well as reducing the reliance on a huge volume of pre-training data (an avenue we are planning to explore in future works). Also, the hierarchical design of CNNs and the fully convolutional aspect of the ResNet50 we adopt allows for efficient processing of large image sizes and supporting variable image sizes/ratios respectively. That being said, many recent vision transformers aim to close these gaps by either incorporating hierarchical priors into their architecture [12] or by directly adapting flat vision transformers for detection tasks [11]. Thus the specific choice of the backbone model could be altered in a future ablation study.

Secondly, and in order to feed in appropriate embeddings to the decoders, that should be able to condition their output based on the current target/query pair, we opt to use a series of cross-attention transformer blocks in order to integrate query features into the target embedding and vice versa. This choice is motivated by the fact that we want the embeddings that are being fed to the decoders to incorporate information from both the target and query embeddings. This differs from the original DETR architecture, that uses simple self-attention blocks for its single input image, as it is trained to detect the same predefined set of classes for all input images.

Finally, after going through a series of cross attention layers, two branches follow. Given that our goal is to perform detection within the target image, we forward the *transformed* embedding of the target image to a pattern spotting decoder that outputs a fixed-size set of predictions (this block is mostly similar to DETR decoder). Note that the pattern spotting decoder classification head only has two classes, as we only care for our model to predict if a particular detection corresponds to our current pattern of interest or not. The second branch forwards the *transformed* embedding of the query image to a sub-image retrieval decoder that outputs a global binary score of whether the query is found within the target image or not.

Thus, our goal with this architecture, named One-Shot DETR or OS-DETR, would now be to train a model that is capable of detecting any query pattern within any target image in a generic fashion, by taking in their images as input and directly predicting the bounding box location of the various occurrences of the query image within the target image, as well as a global confidence score of the presence of the query in the image, the whole in a completely end-to-end fashion. A natural question that arises here, is on what basis will the model detect occurrences? As what constitutes a particular *class* is very task-dependent. Here we argue that our architecture is flexible enough to learn various class definitions,

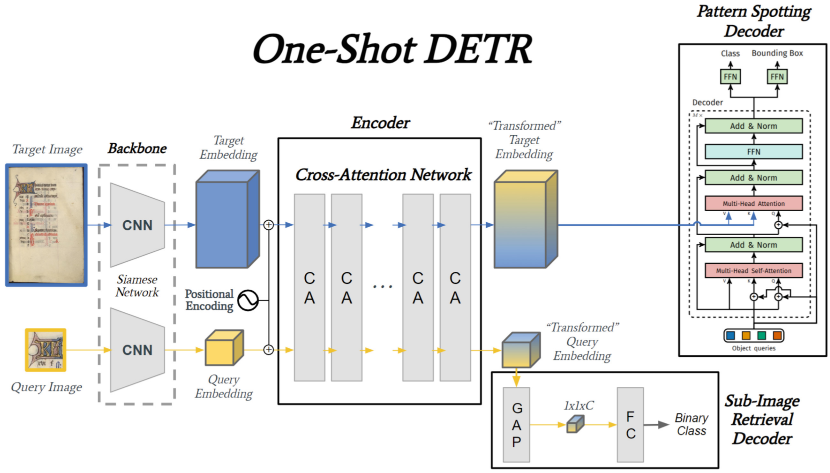


Fig. 2. Taking as input a target/query image pair, both are fed to the same backbone to extract their features. Both embeddings are then fed to a series of cross-attention blocks to compute similarities. The "transformed" target embedding is fed to the pattern spotting decoder that outputs a fixed-size set of detections, whereas the "transformed" query embedding is fed to the sub-image retrieval decoder that outputs a binary class.

which will be implicitly modelled in the training data that is fed to the model. In fact we will show how the same architecture can detect objects either at a semantic level, as required by COCO, or at an instance level, as required by DocExplore, simply by altering the training data.

Loss Term. The loss term for our architecture is essentially the same as the set prediction loss defined in DETR [2], with the addition of a binary cross entropy loss for the sub-image retrieval branch. Our OS-DETR model outputs a fixed-size set of N predictions $\hat{y} = \{\hat{y}_i\}_{i=1}^{N}$ from its pattern spotting branch, on which the Hungarian loss proposed in [2] is computed, and a scalar value $\hat{r} \in [0, 1]$ from its sub-image retrieval branch, on which a binary cross entropy loss is computed. Thus our global loss term is a linear combination between these two loss terms with $\lambda_{Hungarian}, \lambda_{BCE} \in \mathbb{R}$ and set to 1 in our experiments:

$$\mathcal{L}(\{y, r\}, \{\hat{y}, \hat{r}\}) = \lambda_{Hungarian}\mathcal{L}_{Hungarian}(y, \hat{y}) + \lambda_{BCE}\mathcal{L}_{BCE}(r, \hat{r}) \quad (1)$$

3.2 Preliminary Experiments

Preliminary experiments were first conducted on the COCO dataset for the one-shot object detection task in order to assert the validity of our model architecture, before running tests on the sub-image retrieval and pattern spotting tasks on DocExplore dataset, tasks for which we do not have any comparison point in terms of learning-based approaches. This experiment used only the pattern spotting decoder branch of the architecture, as sub-image retrieval is not taken into account in the followed benchmark. The goal of this experiment is to verify that our model architecture proposal stands justified, thus no particular optimization was made for this task.

Implementation Details. As is done in DETR, we use the 4^{th} layer output from the backbone ResNet50. Through testing, 8 cross-attention blocks are used in the encoder and 4 decoder blocks. All transformers have a channel width of 256, a feed forward dimension of 2048, 8 heads as well as a dropout value of 0.1. We train OS-DETR on 4 A100 GPUs, using AdamW optimizer with the learning rate as well as the weight decay set to 10^{-4}. Batch size is set to 16. These same hyper-parameters will be used in all presented experiments.

We follow the same training/evaluation protocol used by other approaches and described in [10] to test our model on the $split_1$ of the COCO dataset. Table 1 shows the result of our model compared to other state of the art approaches on the MS-COCO one-shot object detection benchmark.

Table 1. One-shot object detection performance comparison on the MS-COCO val 2017 dataset (only on $split_1$) in terms of mAP50 score (%).

Method	Seen	Unseen
SiamMask (Arxiv 2018) [14]	38.9	15.3
CoAE (NIPS 2019) [10]	42.2	23.4
AIT (CVPR 2021) [3]	50.1	26.0
OS-DETR (ours)	*39.4*	*20.1*

While our performance is not at the state of the art, it is nevertheless in the *neighbourhood* of good performances on this task, which validates our model architecture and proves that our model is capable of learning useful representations that could be used in the pattern spotting context. Thus, we move to sub-image retrieval and pattern spotting experiments, but before that, we must determine the appropriate data for training our model and better adapt the training to the challenges faced in these tasks. In fact, if we try to directly test this model, trained on COCO for one-shot object detection, on the DocExplore test set following the protocol described in Sect. 6.1, we only achieve ≈3% mAP and ≈0% mAP on sub-image retrieval and pattern spotting respectively, further highlighting the domain and task gaps that needs addressing moving forward.

4 Synthetic Data Generation

As mentioned above, one of the key challenges in this work is the lack of any sort of labelled data that can be used during training. One solution would be to spend time and effort on labelling data for training. This is however much easier said than done, as the manual labelling of pattern spotting data is especially challenging given that, for each pattern, it requires not only annotating its localisation in all images containing it (or more precisely, a near-duplicate occurrence of it), but also be certain of its absence in all other images. And with the large data volume needed for a deep architecture to learn, this approach becomes more unfeasible.

This is why we chose in this work to generate training and validation data synthetically, both as a way to overcome our limitation in this work and to pave the way for other domains that also do not have access to any real training data. Our pattern spotting data synthesis is inspired from [17] that generates synthetic data for the task of co-segmentation in artwork, and from [8] that shows that simple copy/paste is an effective data augmentation for instance segmentation. The basic idea is simple: having access to a collection of background images, as well as to object images, we would randomly sample object images and background images, and paste object images onto background images. This way, we can automatically generate all ground truth data required for training both the pattern spotting as well as the sub-image retrieval branches of our model. For this work, background images were sampled from the historical documents dataset: HORAE [1]. However for object images, and as we do not have access to enough ground-truth labels to extract a meaningful number of diverse query patches from document images, we opted to utilise logo images from LLD [16]. This decision also helps in keeping our proposed approach completely unsupervised. For context, background images have dimensions in the neighbourhood of $1,000px$ whereas pasted logos have dimensions spanning from 25 px all the way up to 300 px, with the majority of the distribution skewed towards the smaller size. Figure 3 presents a visualisation of the synthetic data generation process.

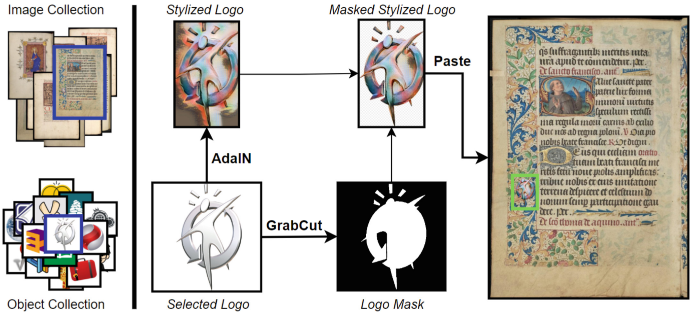


Fig. 3. Visualisation of the synthetic data generation process. Starting with a randomly selected logo (from the object collection), AdaIn applies style transfer and generates a stylized logo, whereas GrabCut generates a binary mask. These two outputs are stacked to generate a masked stylized logo which is then pasted in a random location and scale in a sampled background image (from the image collection).

Synthetic Testing Dataset. Technically speaking, during training, our model is learning tasks of detection and classification from its two branches, however the tasks we are aiming to solve are pattern spotting and sub-image retrieval. The most obvious difference between these two task pairs is the notion of and the importance of ranking in pattern spotting and sub-image retrieval, whereas detection and classification are only concerned with independent input samples. Another big difference is that in pattern spotting there is a high level of negative samples, i.e., the query is not present in the target image, which is not the case in detection training, and certainly not the case in *standard* one-shot object detection. In fact, the one-shot object detection definition, as stated in the literature by [10] (and adopted by follow-up works), assumes that "each feasible target image includes at least one object instance with respect to the class label of the one-shot query", making our tasks more challenging to tackle. For all these reasons, and on top of our synthetic validation set (generated in the same fashion as our training set), we found the need to create a synthetic testing set specifically curated to test the tasks of pattern spotting and sub-image retrieval.

For that, a diverse set of 50 queries were randomly selected along with $1,500$ background images, specifically reserved for testing purposes. For each query, up to 75 background images were randomly selected to paste the query in, with a limit of 10 distinct queries per background and 7 pasted occurrences per query per background (most values are below these limits). This resulted in a diverse and challenging testing set, with over $8,300$ total annotated occurrences distributed in $1,500$ images, that will be crucial for testing our architecture.

5 Training Strategies

Many tricks were implemented in the training of this architecture in order to improve its generalisation capabilities and adaptability to the target tasks.

Negative Query Sampling. As mentioned in the previous section, a big challenge in pattern spotting and sub-image retrieval is the high likelihood of negative samples, i.e., target images that do not contain a given query. For this reason, we decided to implement a negative query sampling probability during training. This is done via a hyperparameter $p_{neg} \in [0, 1]$, and each time an (positive) image pair is selected during training, there is a p_{neg} probability of discarding the query in the current pair, and randomly sampling another query not present in the target image. This teaches the model that not all input pairs will result in a non-empty set of predictions from the pattern spotting branch (greatly reducing false positive predictions), as well as providing samples with $r = 0$ for the proper training of the sub-image retrieval branch.

Random Patch Cropping. One issue with generalisation is that, even though we use historical document images as background images, we still are detecting at the end of the day logos every time a positive pair is fed to the model. So in order to improve generalisation and avoid having the model only detecting logos, or even try and learn to detect only pasted objects, we implement a random patch cropping probability during training. This is done via a hyperparameter $p_{rand} \in [0, 1]$, and each time an image pair is selected during training, there is a p_{rand} probability of discarding the query in the current pair, randomly selecting and cropping an area from the target image, and use that as our query instead. The output goal would then be to detect the area from which the query has been extracted from. The benefits here are two-fold, as (i) this will help in training the model on more diverse data, which is always beneficial, and (ii) this will show samples to the model where the goal is to detect something other than pasted logos, helping in reducing bias in that regard.

Multi-query Approach. In our initial experiments, a single *positive* logo was pasted (multiple times) on a background image, along with some random *noise* logos. Then a random pasted instance of the *positive* logo is cropped and used as query to form a positive pair during training. This was replaced with an approach dubbed **multi-query**. The idea is to select multiple distinct logos and paste each multiple times on the same target image. Then, at training time, any one of these pasted logos can be used as query to form a pair with the target image. Also, and instead of cropping an instance from that target image to use as query, a random instance of that same logo is selected from another target image and cropped from there to be used as query. Beside further enriching and diversifying training data (making the model more robust to background variations in the query), this approach also helped in further conditioning model predictions on current query, as the same target image can now be fed to the model with multiple *positive* queries. This greatly reduced model bias of recognizing *pasted* objects.

Re-ranking. As mentioned above, another key difference between the tasks the model is trained on (detection/classification) and the tasks the model is tested on, is the importance of ranking in pattern spotting/sub-image retrieval. The model, as it is trained, only concerns itself with each input pair individually, and does not have any concept of actually ranking each prediction with respect to other predictions. That is why we implement a re-ranking step that takes in the predicted boxes of the model, and re-rank them based on similarity with the query, using distance between the embeddings of a very light and fast ImageNet pre-trained MobileNet [9]. This allows for better task adaptation, and allows us to better rank exact matches and bring down the slightly more different matches that the model had given a high score to begin with, which is not necessarily a *wrong* behaviour, it is just that other matches are *closer* to the query.

Figure 4 showcases the different ways in which a single target image can be matched to form training pairs.

6 Experimental Results

6.1 DocExplore Benchmark Protocol

We use the DocExplore dataset [6] in order to evaluate and compare our approach to other proposals in the domain. With 1, 500 historical document images

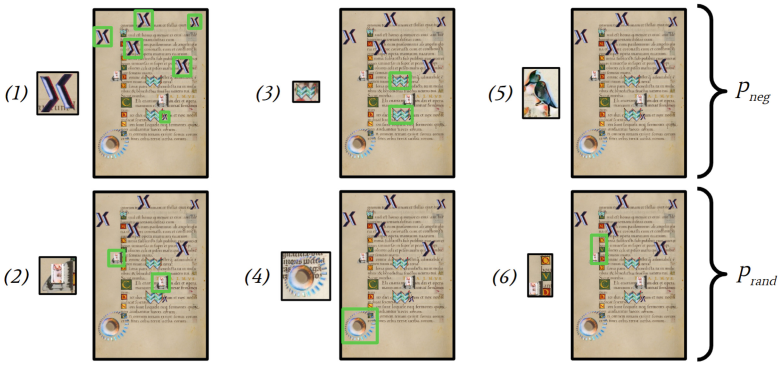


Fig. 4. Possible training pairs for a given target image. Pairs (1) through (4) showcase standard pairing for the 4 unique queries present in the target. Pairs (5) and (6) showcase respectively an example pair for the **negative query sampling** and **random patch cropping**. Green boxes represent the output objective for each pair.

and 1,447 annotated pattern occurrences spanning across 35 unique pattern categories, DocExplore is a unique dataset that allows us to evaluate pattern spotting and sub-image retrieval in the context of mediaeval manuscripts. The metric of choice for both of these tasks is the mean average precision, or mAP, which efficiently compresses ranked lists of predictions into a single scalar value. While for sub-image retrieval counting true positives is straightforward, for pattern spotting an extra step is needed. We employ IoU (Intersection over Union) in order to calculate overlap between predictions and ground-truths. Only predictions with $IoU > 0.5$ with a ground-truth occurrence are considered true positive, as imposed by [6]. It is worth noting that the majority of pattern occurrences have small dimensions (as small as (20 px, 10 px)), which in turn become relatively tiny areas (as small as 0.03%) of the document images that have an average dimension of (600 px, 930 px).

6.2 Pattern Spotting and Sub-image Retrieval

For pattern spotting and sub-image retrieval training/evaluation, more than $220K$ synthetic target images were created, each having multiple possible pair matching during training as showcased in Fig. 4. Due to the high volume of synthetic data generated, our model only needs 20 epochs of training before converging and gaining marginal improvements in the following epochs. After extensive testing, the final model used $p_{rand} = 0.1$ and $p_{neg} = 0.5$, leaving $p_{standard} = 0.4$ for standard target/query pair matching. Also, during training, the ImageNet pre-trained ResNet50 backbone is frozen, as any amount of learning of this backbone led to severe over-fitting on synthetic data.

Table 2 presents an overview of the model performance on the synthetic test set generated following Sect. 4. We can observe how the addition of various strategies significantly boost the model performance, going from 26% mAP and 20%

Table 2. Sub-image retrieval and pattern spotting performance comparison on the synthetic test set (mAP score).

Data	Synthetic Test Set	
Task	Sub-Image Retrieval	Pattern Spotting
Baseline OS-DETR	26.8	20.7
+ p_{rand}	31.7	24.6
+ p_{neg}	75.8	59.3
+ *Multi-Query Approach*	89.8	71.5
+ *Re-Ranking*	**97.0**	**80.2**

mAP respectively in sub-image retrieval and pattern spotting, all the way up to 97% mAP and 80% mAP with the inclusion of all improvements. This showcases our success in approaching both the training and testing tasks, having created a suitable framework for training a sub-image retrieval and pattern spotting system. It is also interesting to notice how much the addition of p_{neg} helped in improving performance, as including negative samples in standard object detection training is generally not useful, due to the fact that negative boxes themselves act as negative samples while training. We analysed that this is most likely due to the transformer nature of the DETR decoder, where all object queries first communicate (via self-attention) before providing the final output. If every input pair results in at least one output box during training, this self-attention between object queries could lead to always having at least one object query that should predict with high confidence the most probable occurrence, even when there is none to be found.

Initial testing on the DocExplore benchmark scored 58% mAP in sub-image retrieval and 22% mAP in pattern spotting, which is a huge improvement when compared to the starting points of 3% mAP and 0% mAP saw in Sect. 3.2. Upon inspection however, we noticed that a large portion of false positives are due to localization imprecision, which is to be expected from such a domain shift. This in turn impacts our re-ranking stage, as we will be computing distances using imprecise crops of patterns. For this reason, and when testing on DocExplore, we opted to adopt a sped-up version of the box re-centering technique used in [7]. This allowed our model performance to go up to an impressive 67% mAP in sub-image retrieval and 38% mAP in pattern spotting, surpassing all previous learning-free approaches with the exception of the recently proposed work of Curi et al. [4]. A comparison between our model and other state of the art models is given in Table 3.

Those results are promising for a multitude of reasons, all revolving around the fact that our approach is learning-based. (i) Being learning-based, our approach opens up a plethora of possible future improvements and refinement both for this specific task, as well as for any other task suffering from scarce to no labelled data. Also, on DocExplore we are competing with learning-free approaches that have been refined and optimised for the past decade on this task.

Table 3. Pattern spotting and sub-image retrieval performance comparison on the DocExplore test set (mAP score). An asterisk (*) denotes a learning-free approach.

Method	Sub-Image Retrieval	Pattern Spotting
En et al. (PR 2016)* [7]	58.0	15.7
Ubeda et al. (PRL 2020)* [19]	57.7	27.2
Curi et al. (ICPR 2022)* [4]	80.3	63.8
OS-DETR (ours)	**67.2**	**38.1**

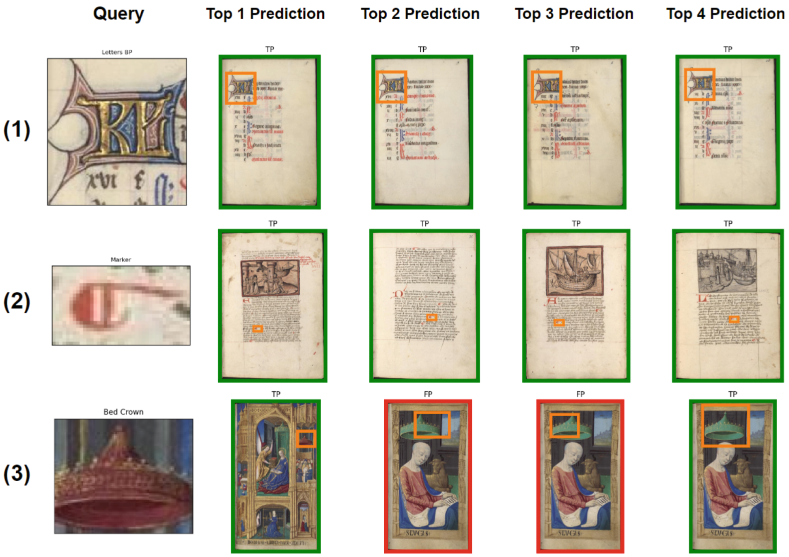


Fig. 5. Top 4 pattern spotting predictions on three DocExplore queries. Rows (1) and (2) showcasing a relatively large (150 px, 160 px) and relatively small (40 px, 20 px) query respectively, whereas row (3) showcases the ability of the model to detect various scales of the same query; a capability not attainable for previous learning-free works.

So it would be reasonable to predict a similar, if not more important, improvement in following versions of our first of a kind approach. (ii) These experiments highlight the flexibility of our proposed framework, as the same model is able to both perform one-shot object detection on MS-COCO benchmark when trained on COCO data, while also being able to perform pattern spotting on DocExplore benchmark when trained on synthetically generated data. This point is crucial, as user-intent and task-objective can change drastically in real world applications. Being able to control the final behaviour of the model by modelling the distribution of synthetic training data is a very promising prospective. (iii) Our approach can perform tasks that are simply not possible with other learning-free approaches, such as detecting occurrences of patterns with a high scale variability. An example of that can be seen in Fig. 5 at row (3). (iv) Finally, our proposed framework is very flexible and can most definitely be improved in upcoming iterations, with the possibility to swap the backbone for more expressive or even more task-adequate backbones, to improve the synthetic data generation process, mainly improving the pasting process of the queries, and many other aspects that can be easily swapped to improve performance.

7 Conclusion

We have demonstrated the feasibility of a learning-based approach to solve sub-image retrieval and pattern spotting tasks, whose performance closely approaches the decade-long optimised learning-free methods on the DocExplore benchmark. A novel architecture, OS-DETR, has been proposed that is capable of adapting to various task-definitions. A flexible and parameterizable synthetic data generation process has been described, showing promising prospective for domains with scarce to no labelled data. Finally, a set of training strategies have been proposed that significantly close the gap between classical detection/classification training and sub-image retrieval/pattern spotting testing. Future work will focus on improving the generalisation capability of our framework, the synthetic data generation process, as well as establishing better benchmarks to evaluate said improvements.

Acknowledgements. The first author is supported by the French National Research Agency (ANR) and Region Normandie under grant HAISCoDe. This work was performed using HPC resources from GENCI-IDRIS (Grant 2023-AD011013677R1).

References

1. Boillet, M., Bonhomme, M.L., Stutzmann, D., Kermorvant, C.: HORAE: an annotated dataset of books of hours. In: Proceedings of the 5th International Workshop on Historical Document Imaging and Processing, pp. 7–12. ACM, Sydney (2019). https://doi.org/10.1145/3352631.3352633
2. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12346, pp. 213–229. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58452-8_13
3. Chen, D.J., Hsieh, H.Y., Liu, T.L.: Adaptive image transformer for one-shot object detection. In: CVPR 2021, pp. 12247–12256 (2021). https://doi.org/10.1109/cvpr46437.2021.01207
4. Curi, Z., Nicolas, S., Tranouez, P., Britto, A.D.S., Heutte, L.: Image retrieval and pattern spotting on historical documents with binary descriptors. In: ICPR 2022, pp. 3893–3899. IEEE, Montreal (2022). https://doi.org/10.1109/ICPR56361.2022.9956416
5. Dai, Z., Cai, B., Lin, Y., Chen, J.: UP-DETR: unsupervised pre-training for object detection with transformers. In: CVPR 2021, pp. 1601–1610 (2021). https://doi.org/10.1109/cvpr46437.2021.00165

6. En, S., Nicolas, S., Petitjean, C., Jurie, F., Heutte, L.: New public dataset for spotting patterns in medieval document images. J. Electron. Imaging **26**(1), 011010 (2016). https://doi.org/10.1117/1.JEI.26.1.011010
7. En, S., Petitjean, C., Nicolas, S., Heutte, L.: A scalable pattern spotting system for historical documents. Pattern Recogn. **54**, 149–161 (2016). https://doi.org/10.1016/j.patcog.2016.01.014
8. Ghiasi, G., et al.: Simple copy-paste is a strong data augmentation method for instance segmentation. In: CVPR 2021, pp. 2917–2927. IEEE, Nashville (2021). https://doi.org/10.1109/CVPR46437.2021.00294
9. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications. CoRR abs/1704.04861 (2017). https://doi.org/10.48550/ARXIV.1704.04861. arXiv Version Number: 1
10. Hsieh, T.I., Lo, Y.C., Chen, H.T., Liu, T.L.: One-shot object detection with co-attention and co-excitation. In: Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F.d., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 32. Curran Associates, Inc. (2019). https://doi.org/10.48550/arXiv.1911.12529
11. Li, Y., Mao, H., Girshick, R., He, K.: Exploring plain vision transformer backbones for object detection (2022). https://doi.org/10.48550/ARXIV.2203.16527. arXiv Version Number: 2
12. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: ICCV 2021, pp. 10012–10022 (2021). https://doi.org/10.1109/iccv48922.2021.00986
13. Lv, W., et al.: DETRs beat YOLOs on real-time object detection (2023). https://doi.org/10.48550/ARXIV.2304.08069. arXiv Version Number: 2
14. Michaelis, C., Ustyuzhaninov, I., Bethge, M., Ecker, A.S.: One-shot instance segmentation. CoRR abs/1811.11507 (2018). https://doi.org/10.48550/arXiv.1811.11507
15. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2017). https://doi.org/10.1109/TPAMI.2016.2577031
16. Sage, A., Timofte, R., Agustsson, E., Gool, L.V.: Logo synthesis and manipulation with clustered generative adversarial networks. In: CVPR 2018, pp. 5879–5888. IEEE, Salt Lake City (2018). https://doi.org/10.1109/CVPR.2018.00616
17. Shen, X., Efros, A.A., Joulin, A., Aubry, M.: Learning co-segmentation by segment swapping for retrieval and discovery. arXiv (2021). https://doi.org/10.48550/arXiv.2110.15904
18. Zhang, G., Luo, Z., Cui, K., Lu, S., Xing, E.P.: Meta-DETR: image-level few-shot detection with inter-class correlation exploitation. IEEE Trans. Pattern Anal. Mach. Intell. 1–12 (2022). https://doi.org/10.1109/TPAMI.2022.3195735
19. Úbeda, I., Saavedra, J.M., Nicolas, S., Petitjean, C., Heutte, L.: Improving pattern spotting in historical documents using feature pyramid networks. Pattern Recogn. Lett. **131**, 398–404 (2020). https://doi.org/10.1016/j.patrec.2020.02.002

Zero-Shot Prompting and Few-Shot Fine-Tuning: Revisiting Document Image Classification Using Large Language Models

Anna Scius-Bertrand[1,2](✉), Michael Jungo[1,2], Lars Vögtlin[2], Jean-Marc Spat[1], and Andreas Fischer[1,2]

[1] University of Applied Sciences and Arts Western Switzerland, Delémont, Switzerland
{anna.scius-bertrand,michael.jungo,jean-marc.spat,andreas.fischer}@hefr.ch
[2] University of Fribourg, Fribourg, Switzerland
{anna.scius-bertrand,michael.jungo,lars.vogtlin,andreas.fischer}@unifr.ch

Abstract. Classifying scanned documents is a challenging problem that involves image, layout, and text analysis for document understanding. Nevertheless, for certain benchmark datasets, notably RVL-CDIP, the state of the art is closing in to near-perfect performance when considering hundreds of thousands of training samples. With the advent of large language models (LLMs), which are excellent few-shot learners, the question arises to what extent the document classification problem can be addressed with only a few training samples, or even none at all. In this paper, we investigate this question in the context of zero-shot prompting and few-shot model fine-tuning, with the aim of reducing the need for human-annotated training samples as much as possible.

Keywords: Document Image Classification · OCR · Deep Learning · Transformers · Large Language Models · Few-Shot Learning · Prompting

1 Introduction

The RVL-CDIP dataset, introduced by Harley et al. [6], is a subset of 400 000 labeled document images derived from the IIT-CDIP collection, originating from a litigation against the tobacco industry [11], which has significantly boosted the exploration of deep learning methods for document image classification in the last decade. Notable advancements include the application of convolutional neural networks [18], the integration of text, image, and layout embeddings as demonstrated by LayoutLM [20], OCR-free document understanding through Transformers (Donut) [10], and cross-modal strategies that fuse image and textual analysis techniques [2]. The cross-modal strategies stand out by achieving an impressive classification accuracy of 97.05% across 16 distinct document types, including letters, forms, and emails.

A. Scius-Bertrand and M. Jungo—These authors contributed equally to this work.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 152–166, 2025.
https://doi.org/10.1007/978-3-031-78495-8_10

Nevertheless, neural network models with a rising number of trainable parameters require a large training set of labeled documents to perform satisfactorily. For example, the RVL-CDIP benchmark needs a training set of 320 000 labeled documents to distinguish between 16 document classes. If the document categories change, or for a new dataset, the training set must be relabeled accordingly, which leads to a costly and time-consuming human effort.

The recent advent of large language models (LLMs) has impressively shown that large networks with billions, or even more than a trillion, parameters, only need very few training samples, if any, to solve challenging tasks in natural language processing, including closed-book question answering, translation, and reading comprehension [3]. LLMs typically rely on unsupervised pre-training of decoder-only transformer architectures on a large body of texts from the internet, such as the Common Crawl dataset [17], followed by reinforcement learning from human feedback, to achieve astounding generalization capabilities.

With respect to the task of document classification, the question arises to what extent LLMs may be capable of solving the task without the need of hundreds of thousands of learning samples, relying on their text understanding capabilities of the document texts, which are extracted by means of optical character recognition (OCR).

In the present paper, we explore this question in a comprehensive benchmark evaluation that takes into account several state-of-the art LLMs for text analysis (Mistral [8], GPT-3 [3], GPT-4 [1]), defines different training scenarios, and puts the LLM results into a broader context by including also a selection of smaller language models (RoBERTa [12]), text embedding models (Jina [5]), OCR-free image-based models (Donut [10]), and multi-modal LLMs (GPT-4-Vision [1]) in the comparison. The aim of the benchmark evaluation is to investigate the document classification performance for an increasing number of learning samples, starting with zero-shot prompting, where only a textual description of the task is provided to the model, and ending with few-shot model fine-tuning using 100 samples per class. Note that fine-tuning of LLMs is a challenging task on standard hardware. We rely on Low-Rank Adaptation (LoRA) [7] to fine-tune one of the smaller open source LLMs, Mistral-7B [8], for the purpose of our benchmark.

No new method is proposed in this paper. Instead, our contributions are:

- A comprehensive benchmark[1] for evaluating document classification in a few-shot training scenario.
- Comparing LLM prompting vs LLM fine-tuning for document classification.
- Comparing generative vs embedding-based document classification.
- Comparing text-based vs image-based document classification.

With this benchmark evaluation and comparisons of current interest, we aim to inspire and support further research on few-shot document classification.

The paper is structured as follows: An introduction of the dataset is given in Sect. 2, a description of the methods in Sect. 3, and in Sect. 4 the experimental results are presented. Lastly, we draw some conclusions and discuss future work.

[1] Available at: https://github.com/asciusb/LLM-CDIP.

2 Data

The RVL-CDIP dataset [6] contains 25 000 scanned grayscale images per document class for 16 classes: letter, form, email, handwritten, advertisement, scientific report, scientific publication, specification, file folder, news article, budget, invoice, presentation, questionnaire, resume, and memo. Several examples from different categories are shown in Fig. 1.

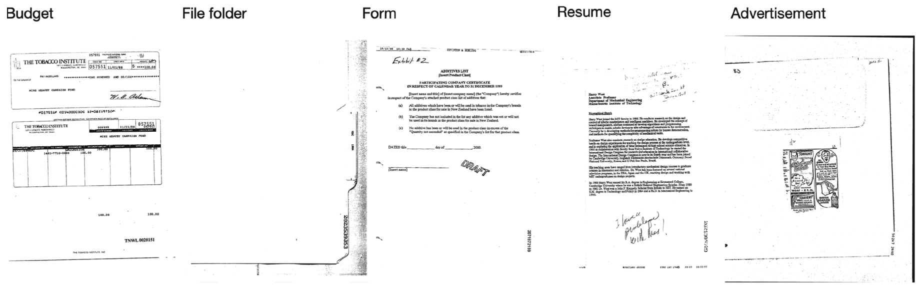


Fig. 1. Example document images from the RVL-CDIP dataset.

Many document classification and understanding methods require the text contained in the images, which can be obtained with any OCR engine. No matter how good the OCR engines have become, there are still difficulties to obtain a perfect textual representation of the documents. Most notably, parts of some images are illegible due to degradation or any other quality issues. This makes purely text-based methods more difficult, as they also need to deal with the noise created by the OCR.

Furthermore, several categories contain images with very little textual information, making it potentially more difficult to distinguish between them as file folders or advertisements shown in Fig. 1. There is also a mix between handwriting and printed text in the documents, which could cause issues for the OCR. In any case, the visual information that is attached to the different types of writing is lost in the process. Therefore, any OCR-based method relies much more heavily on the meaning of the content, rather than its structure, although the grammatical structure remains a significant part of the textual representation.

3 Methods

In the following, we describe the different methods used in the benchmark: Text-based classification using generative LLMs, text embedding classification, image-based document classification, and multi-modal LLMs.

3.1 Text-Based Classification

OCR. For OCR, we use Amazon's Textract, which performed reasonably well on a few example images from the RVL-CDIP dataset, especially with respect to low-resolution, skewed documents, and handwritten elements. The resulting machine-readable text follows a natural reading order and the text lines are delimited by line-break characters.

LLMs. We focus on some of the best-performing LLMs from the current state of the art, including the GPT models from OpenAI [1,3,15,16] and the models from Mistral AI [8]. The transformer-based models have been pretrained on texts from the internet and fine-tuned to follow instructions, as well as using reinforcement learning from human feedback.

Table 1. LLM versions.

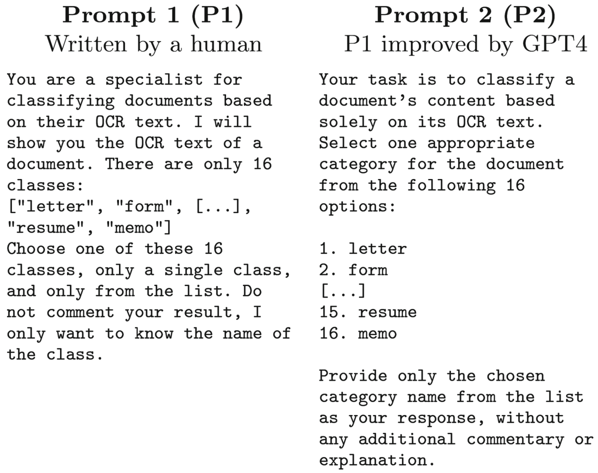
Model	Version
Mistral-7B	Open source [8]
Mixtral-8x7B	Open source [9]
Mistral-Medium	mistral-medium-2312
Mistral-Large	mistral-large-2402
GPT-3.5	gpt-3.5-turbo-0125
GPT-4	gpt-4-turbo-2024-04-09

Table 1 lists the specific versions used. Mistral-7B has 7×10^9 parameters [8], Mixtral-8x7B has 47×10^9 [9], and GPT-3 has 175×10^9 [3]. To the best of our knowledge, no official information is available at the moment for the larger models.

LLM Prompts. The LLMs are used in chat completion mode with a system prompt that specifies the classification task in natural language. The prompt was first formulated by a human. Afterwards, we used GPT-4 in chat completion mode to refine the original prompt and used it again to further tweak the refined prompt. The three resulting prompts are shown in Fig. 2.

Zero-Shot Prompting: Only the system prompt is used, relying on the semantics of the category names for classification.

One-Shot Prompting: We provide context to the LLM by including 16 additional pairs of prompts $(user_i, assistant_i)$ after the system prompt, one for each category, where $user_i$ is an OCR text from the training set and $assistant_i$ is its category name.



Prompt 1 (P1)
Written by a human

```
You are a specialist for
classifying documents based
on their OCR text. I will
show you the OCR text of a
document. There are only 16
classes:
["letter", "form", [...],
"resume", "memo"]
Choose one of these 16
classes, only a single class,
and only from the list. Do
not comment your result, I
only want to know the name of
the class.
```

Prompt 2 (P2)
P1 improved by GPT4

```
Your task is to classify a
document's content based
solely on its OCR text.
Select one appropriate
category for the document
from the following 16
options:

1. letter
2. form
[...]
15. resume
16. memo

Provide only the chosen
category name from the list
as your response, without
any additional commentary or
explanation.
```

Prompt 3 (P3)
P2 tweaked with GPT4

```
Classify the content of a
document using its OCR text.
Choose exactly one category
from the list below that best
fits the document:

- letter
- form
[...]
- resume
- memo

Respond with the name of
the selected category only.
Do not provide additional
comments or explanations.
```

Fig. 2. Three system prompts considered for document classification. Note that the list of 16 categories has been shortened in the figure, indicated by [. . .], to save space.

Fine-Tuning LLMs. LLMs have learned a deep understanding of text through an extensive pretraining. Even though their primary appeal is to use them generatively, which makes them extremely flexible, it would stand to reason that their capabilities can also serve as a base for a classification model. In particular, understanding documents in a low-resource context could benefit from their vast knowledge of all kinds of texts, including a large variety of documents. Hence, we explore the task of document classification by adding a classifier on top of an LLM. In order to fine-tune the LLM, we further employ Low-Rank Adaptation (LoRA) [7], to adapt it to the specific documents at hand.

As a comparison, to evaluate the effectiveness of using an LLM as the base model, we also fine-tune the largest available pretrained RoBERTa [12], which only has 355M parameters. Because of its comparatively small size, the model is fine-tuned fully instead of having to resort to LoRA adapters.

Finally, to see whether adding a classifier head is really necessary, we also fine-tune the same model in a generative manner, where at the end of each document, a classification instruction is added. With the generative approach, the model maintains the ability to be adapted to any kind of output, as opposed to the classifier, where the output is limited to the classes it has been trained for. There are a few disadvantages that come with this flexibility, most notably that the output is no longer guaranteed to be exactly the class that was asked for, and an increase in compute, as the model now needs to predict multiple tokens. Even when the class name consists of a single token, the generative model needs to predict at least an end-of-sequence token in addition to the class name, which requires multiple forward passes.

3.2 Embedding-Based Methods

In addition to generative approaches for classifying OCR texts, we also consider a standard classification setup based on feature vector representations of the OCR texts in an embedding space with a k-nearest neighbor classification (KNN). We also conducted preliminary experiments with multi-layer perceptions (MLP), the results of which were close to the KNN but never outperformed it.

We focus on some of the best-performing text embedding models from the current state of the art, specifically a selection of models proposed by Jina AI [5], Mistral AI, and OpenAI [13]. The versions used are listed in Table 2.

Table 2. Embedding model versions.

Model	Version	Embedding size
Jina-v2	jina-embeddings-v2-base-en	768
Mistral-embed	mistral-embed	1024
OpenAI-small	text-embedding-3-small	1536
OpenAI-large	text-embedding-3-large	3072

3.3 Image-Based Methods

Donut. Donut is a Transformer-based model for Visual Document Understanding (VDU) that operates entirely on images, without having to rely on the OCR. Besides a large number of synthetic documents, the pretraining also included the full IIT-CDIP dataset, which makes it particularly well-suited for classifying documents from the RVL-CDIP subset.

In [10], they already fined-tuned Donut on various downstream tasks, including the classification of RVL-CDIP, which showed excellent results. However, this was conducted on the complete RVl-CDIP dataset with 320K images in the training set, whereas we are investigating whether it also performs well when the training data is severely limited.

3.4 Multi-modal Methods

GPT-4-Vision. Recently, GPT-4-Vision was introduced as a multi-modal extension of GPT-4, which accepts a combination of text and/or image inputs from the user [1] (with the same model version as indicated in Table 1).

Image-Based Zero-Shot Prompting. The first sentence of the system prompts listed in Fig. 2 is changed to “Your task is to classify a document.” without mentioning the OCR text. Only the document image is provided as input.

Bimodal Zero-Shot Prompting. The first sentence of the system prompts is changed to “Your task is to classify a document based on its OCR text and scanned image, which are both provided by the user.” Afterwards, both the OCR text and the document image are provided as input.

4 Experimental Evaluation

In this section, we first describe the experimental setup of the benchmark, followed by results obtained for prompting, embedding, and fine-tuning, respectively. At the end, all results are summarized and put into context with other results from the current state of the art.

4.1 Setup

Training. For zero-shot and one-shot prompting, we consider zero samples and 16 samples (one per class), respectively. For few-shot fine-tuning, we investigate an increasing number of 160 samples (ten per class), 800 samples (50 per class), and 1 600 samples (100 per class), which are randomly chosen from the original RVL-CDIP training set. Note that the smaller training sets are included in the larger ones.

Validation. For optimizing hyperparameters, we consider a validation set of an additional 160 samples (ten per class), which are randomly chosen from the original RVL-CDIP validation set.

Testing. We define a test scenario **RVL-CDIP-160x5**, which consists of five random selections (without overlap) of 160 samples (ten per class) from the original test set, for which we obtain a high-quality OCR. We report the mean accuracy and standard deviation over the five test sets. As demonstrated in the experiments (see Sect. 4.4), this selection results in a representative subset, which reduces the cost of testing significantly when using the APIs of Mistral AI and OpenAI for their LLMs, as they are not freely available.

Additionally, the original **RVL-CDIP-40K** test set, which contains 40 000 samples, is also included in the evaluation, whenever possible. Having this large test set available, increases the confidence in the perceived evaluation and serves as a reference to existing results from the literature.

OCR. For the training sets, validation set, and the RVL-CDIP-160x5 test sets, we extract the text from the images by using Amazon’s Textract. As running the OCR on the full RVL-CDIP-40K test set goes beyond the scope of this research, we fall back to the original OCR texts from the IIT-CDIP collection [11]. The original OCR was performed with a 90 s-era OCR engine, which unquestionable produced a lower quality output. Nevertheless, we include it to judge how representative our randomly selected subsets are with respect to an established test set of considerable size.

Models Setup. For zero-shot and one-shot prompting, we rely on the APIs of Mistral AI and OpenAI respectively, specifying a temperature of 0 to encourage precise, non-creative answers. The LLM versions used are indicated in Table 1.

For KNN-based classification of OCR embeddings, we use the APIs of Jina AI, Mistral AI, and OpenAI to extract the embedding vectors. The embedding models considered are indicated in Table 2. The parameter $k \in \{1, 3, 5, 7, 9\}$ and the metric (Euclidean or cosine) are optimized with respect to the classification accuracy on the validation set.

Fine-tuning Donut is performed by using the official implementation[2] and with the suggested configuration of the CORD dataset [14]. In particular, the image size is fixed to 1280 × 960 pixels, which provides a good compromise between readability for the human eye and computational effort for the GPU.

Mistral-7B [8] is used as the base model for all experiments for the LLM fine-tuning. As proposed in QLoRA [4], the base model is quantized into 4-bit weights and trainable LoRA adapters are added to all linear layers. We chose rank $r = 8$ with $alpha = 16$ based on preliminary experiments on the validation set. On the other hand, for RoBERTa, all parameters are fine-tuned, since the model is small enough to be trained fully in a reasonable time on the available hardware.

The generative fine-tuning adds a classification directive at the end of each document, specifically "`### Classification:`" followed by the class name to be predicted. Solely the tokens after the classification directive contribute to the loss and therefore the weight updates. This is not an instruction tuning, meaning that the model does not receive the instructions that were used for the one-shot prompting.

RoBERTa and Mistral-7B are both implemented and trained with HuggingFace's `transformers` [19] library and `bitsandbytes`[3] to support QLoRA.

4.2 Prompting Results

In a preliminary experiment on the validation set, we optimized the LLM system prompt (see Fig. 2): Written by a human (P1), enhanced by GPT-4 (P2), and enhanced twice by GPT-4 (P3). For a few-shot learning task with GPT-3.5 using 32 training samples (two per class) and evaluating on 48 validation samples (three per class), the system prompt P2 achieved the best accuracy (60.4%), outperforming P1 (58.3%) and P3 (56.2%). That is, an enhancement of the human prompt by GPT-4 was beneficial and P2 was selected for all subsequent experiments.

Table 3 shows the prompting results on the test sets of RVL-CDIP-160x5. For zero-shot prompting, the mean classification accuracy ranges from 25.0% (Mixtral-8x7B) to 69.9% (GPT-4-Vision), highlighting a large variability among the models.

[2] https://github.com/clovaai/donut.

[3] https://github.com/TimDettmers/bitsandbytes.

Table 3. Zero-shot and one-shot prompting. Document classification results on the RVL-CDIP-160x5 test sets, indicating the number of training samples (#Train), as well as the mean and standard deviation of the classification accuracy (%) and invalid answers (%) across the five subsets. The best results per training scenario are marked in **bold**.

#Train	Input	Model	Accuracy	Invalid
0	OCR	Mistral-7B	45.4 ± 2.8	17.0 ± 2.9
0	OCR	Mixtral-8x7B	25.0 ± 2.9	56.2 ± 1.7
0	OCR	Mistral-Medium	54.6 ± 4.1	6.4 ± 3.1
0	OCR	Mistral-Large	54.4 ± 4.1	14.6 ± 1.6
0	OCR	GPT-3.5	36.9 ± 1.1	32.1 ± 1.5
0	OCR	GPT-4	61.8 ± 2.0	2.1 ± 1.2
0	Image	GPT-4-Vision	**69.9 ± 2.0**	0.5 ± 0.5
0	OCR+Image	GPT-4-Vision	69.4 ± 1.7	0.8 ± 0.7
16	OCR	Mistral-7B	47.1 ± 3.7	22.9 ± 2.8
16	OCR	Mixtral-8x7B	48.2 ± 5.9	13.4 ± 3.3
16	OCR	GPT-3.5	**58.8 ± 2.1**	4.6 ± 1.8

GPT-4-Vision significantly outperforms GPT-4, highlighting the importance of the document image for classification. Bimodal prompting with both OCR text and image did not further improve the results when compared with image-only prompting. Note, however, that GPT-4-Vision is capable of performing OCR, at least implicitly, when the input consists only of the document image.

One of the main limitations of the smaller models (Mistral-7B, Mixtral-8x7B, GPT-3.5) are invalid answers. Instead of only responding with a category name, as requested in the prompt, the models tend to produce longer responses, e.g. “The text provided appears to be a notice for a membership investment in the Florida Retail Political Action Committee”. We do not post-process the responses, thus any deviation from valid category names is considered invalid.

The one-shot prompting results show that the mean accuracy can be improved for the smaller models (Mistral-7B, Mixtral-8x7B, GPT-3.5) by providing one example per class, in particular Mixtral-8x7B is improved from 25.0% to 48.2% while reducing the number of invalid answers from 56.2% to 13.4%.

We did not test one-shot prompting for the larger models due to the large number of tokens that are added to the prompt (16 OCR texts or images, respectively), which significantly increases the costs of using the commercial APIs.

4.3 Embedding Results

The results for KNN-based classification of OCR text embeddings are shown in Table 4. The mean accuracy achieved by Mistral-embed, OpenAI-small, and OpenAI-large are fairly similar and outperform most of the LLM prompting results when 800 or more training samples are considered, demonstrating that embeddings are a promising strategy for a few-shot learning scenario.

Table 4. KNN classification of OCR text embeddings. Document classification results on the RVL-CDIP-160x5 test sets, indicating the number of training samples (#Train), as well as the mean and standard deviation of the classification accuracy (%) across the five subsets. The best results per training scenario are marked in **bold**.

#Train	Input	Embedding	Accuracy
160	OCR	Jina-v2	41.9 ± 1.6
160	OCR	Mistral-embed	**56.4** ± **3.0**
160	OCR	OpenAI-small	53.9 ± 2.9
160	OCR	OpenAI-large	54.4 ± 1.2
800	OCR	Jina-v2	52.9 ± 4.0
800	OCR	Mistral-embed	63.5 ± 2.0
800	OCR	OpenAI-small	62.4 ± 3.2
800	OCR	OpenAI-large	**64.8** ± **3.1**
1 600	OCR	Jina-v2	57.1 ± 4.5
1 600	OCR	Mistral-embed	66.8 ± 2.7
1 600	OCR	OpenAI-small	65.8 ± 3.4
1 600	OCR	OpenAI-large	**67.8** ± **3.8**

There is one exception: Zero-shot prompting using GPT-4-Vision (69.9% mean accuracy) outperforms all embedding models tested, even when using 1 600 training samples.

A visualization of the OpenAI-large embeddings using t-SNE dimensionality reduction is depicted in Fig. 3, illustrating the capability of the embedding model to form class-wise clusters based on the OCR text.

4.4 Fine-Tuning Results

Table 5 reports the results for fine-tuning RoBERTa, Mistral-7B, and Donut in the different few-shot training scenarios. All fine-tuned models outperform LLM prompting and embedding strategies from the previous sections when 800 training samples or more are used. The results also indicate that almost no invalid responses are generated by the fine-tuned models.

One model stands out with an excellent performance for few-shot learning: Generative fine-tuning of Mistral-7B. Even when providing only 160 training

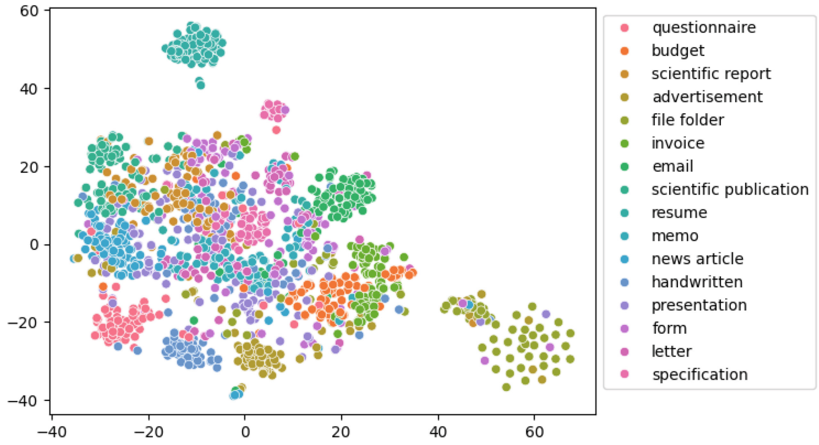


Fig. 3. Embedding of the 1 600 training samples using the OpenAI-large model, visualized with t-SNE.

samples (10 samples per class), the LLM achieves a promising mean accuracy of 72.5% on the RVL-CDIP-160x5 test sets, creating a noticeable gap to the next best model (RoBERTa, 59.8%). Classification fine-tuning is less successful when considering only 160 training samples but catches up for 800 and more, achieving the overall best result of 83.4% for 1 600 training samples.

Besides the aforementioned results, Table 5 also contains the results for the original RVL-CDIP-40K test set. Regarding the OCR-based models, the results show a systematic decrease in accuracy for RVL-CDIP-40K when compared to RVL-CDIP-160x5. This is most likely due to the rather low OCR quality of the original dataset. In contrast, we are considering state-of-the-art OCR results for RVL-CDIP-160x5.

This hypothesis is further strengthened when taking the OCR-free (Donut) results into account, which exhibit a much smaller difference between the two test scenarios. Concretely, for image-based classification with Donut the RVL-CDIP-40K test results are within one or two standard deviations of the RVL-CDIP-160x5 test results for all three training scenarios (160, 800, 1 600). Therefore, we conclude that the five-fold selection of 160 test samples is, indeed, representative for evaluating the RVL-CDIP classification challenge.

Even though the Mistral-7B-Class achieved the single highest result on the RVL-CDIP-160x5 test sets, the generative model (Mistral-7B-Gen) is much more consistent across multiple scenarios. Given that the generative model retains its flexibility, it makes it even more appealing than using a model with a classifier head.

Table 5. Few-shot model fine-tuning. Document classification results on the RVL-CDIP-160x5 and RVL-CDIP-40K test sets, indicating the number of training samples (#Train) and the classification accuracy (%) across the five subsets. The best results per training scenario are marked in **bold**.

#Train	Input	Model	RVL-CDIP-160x5		RVL-CDIP-40K	
			Accuracy	Invalid	Accuracy	Invalid
160	OCR	RoBERTa	59.8 ± 3.8	0.0 ± 0.0	50.2	0.0
160	OCR	Mistral-7B-Class	51.1 ± 4.0	0.0 ± 0.0	23.4	0.0
160	OCR	Mistral-7B-Gen	**72.5 ± 3.9**	0.4 ± 0.3	66.7	0.2
160	Image	Donut	42.8 ± 3.0	0.8 ± 0.7	44.2	1.2
800	OCR	RoBERTa	74.9 ± 4.9	0.0 ± 0.0	66.8	0.0
800	OCR	Mistral-7B-Class	78.3 ± 2.4	0.0 ± 0.0	58.7	0.0
800	OCR	Mistral-7B-Gen	**79.5 ± 3.3**	0.5 ± 0.5	72.8	0.2
800	Image	Donut	70.1 ± 2.6	0.1 ± 0.2	71.4	0.2
1 600	OCR	RoBERTa	78.0 ± 2.0	0.0 ± 0.0	69.3	0.0
1 600	OCR	Mistral-7B-Class	**83.4 ± 4.3**	0.0 ± 0.0	66.6	0.0
1 600	OCR	Mistral-7B-Gen	82.4 ± 2.1	0.5 ± 0.3	74.7	0.3
1 600	Image	Donut	73.8 ± 1.9	0.0 ± 0.0	76.4	0.1

4.5 Results Summary

A summary of the classification results is given in Table 6 with the best models for each training scenario (zero-shot prompting or few-shot learning) and input modality (OCR text and/or image). In the case of OCR text, we include both the embedding approach and end-to-end models. The results achieved on the RVL-CDIP-160x5 test sets are put into context with the results reported in the literature on RVL-CDIP-40K that use the full 320 000 documents for the training.

For zero-shot prompting, the largest LLMs from OpenAI, in particular the multi-modal GPT-4-Vision model, demonstrate an impressive generalization capability with a mean accuracy of 69.9% on the test sets, considering the fact that in this scenario the document classes can be changed on the fly, without the need to annotate learning samples.

Regarding the fine-tuning, the smaller Mistral-7B model stands out in its capability to rapidly adapt to the classification task with only very few training samples when using the generative fine-tuning based on LoRA. Fine-tuning with ten samples per class leads to a mean accuracy of 72.5%.

When fine-tuning with 100 samples per class, which is still considered to be a small amount of training data for the document classification task, the fine-tuned Mistral-7B model with a classifier head achieves the overall best mean accuracy of 83.4%. This is a notable achievement when compared to the 85.0% accuracy reported in [2] for a fully trained BERT model using 320 000 training samples. Admittedly, Mistral-7B is an order of magnitude larger than BERT in

Table 6. Summary of document classification results. The best-performing approach is listed for each zero-shot to few-shot training scenario evaluated on the RVL-CDIP-160x5 test sets. The results are put into context with fully trained models evaluated on the RVL-CDIP-40K test set.

#Train	Input	Model	Accuracy
0	OCR	GPT-4	61.8 ± 2.0
0	Image	GPT-4-Vision	**69.9** ± **2.0**
0	OCR+Image	GPT-4-Vision	69.4 ± 1.7
160	OCR	Mistral-embed+KNN	56.4 ± 3.0
160	OCR	Mistral-7B-Gen	**72.5** ± **3.9**
160	Image	Donut	42.8 ± 3.0
800	OCR	OpenAI-large+KNN	64.8 ± 3.1
800	OCR	Mistral-7B-Gen	**79.5** ± **3.3**
800	Image	Donut	70.1 ± 2.6
1 600	OCR	OpenAI-large+KNN	67.8 ± 3.8
1 600	OCR	Mistral-7B-Class	**83.4** ± **4.3**
1 600	Image	Donut	73.8 ± 1.9
320 000	OCR	BERT [2]	85.0
320 000	Image	Donut [10]	95.3
320 000	OCR+Image	BERT+NasNet [2]	97.1

terms of parameters, nevertheless, it indicates that very promising results can be achieved with much less training data.

5 Conclusion

The RVL-CDIP dataset was originally proposed as a document classification challenge using hundreds of thousands of training samples. Under these conditions, the state of the art has gradually achieved near-perfect performance. By revisiting the question of document classification under the perspective of considering only very few training samples (or none at all), this paper investigates the capacity of current document models to rapidly generalize to new tasks.

We contribute a comprehensive set of benchmark results that explore the question with prompts, embeddings, and model fine-tuning using methods from the current state of the art for image and text analysis. We demonstrate the feasibility of zero-shot and few-shot document classification using LLMs, achieving results that, although promising between 69.9% and 83.4% mean accuracy, leave significant room for improvement.

An important line of future research is related to document foundation models. The strongest few-shot fine-tuning results reported in this paper were achieved with large text-based models (Mistral-7B). However, the state of the art clearly demonstrates that combining image and text leads to significantly better results for fully trained models. Therefore, integrating more visual information into document foundation models is expected to significantly improve few-shot document classification. There is an increasing number of multi-modal LLMs [21] that may be investigated in this context.

Other lines of research include improvements of the prompts, e.g. by providing additional semantics to the LLM that go beyond only the name of the category. Finally, it would be beneficial to explore different learning strategies for unlabeled data, including self-training and unsupervised contrastive learning, to further improve the results of few-shot fine-tuning.

References

1. Achiam, J., et al.: GPT-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Bakkali, S., Ming, Z., Coustaty, M., Rusiñol, M.: Visual and textual deep feature fusion for document image classification. In: Proceedings of the International Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 562–563 (2020)
3. Brown, T., et al.: Language models are few-shot learners. In: Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 1877–1901 (2020)
4. Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. QLoRA: efficient finetuning of quantized LLMs. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
5. Günther, M., et al.: Jina embeddings 2: 8192-token general-purpose text embeddings for long documents. arXiv preprint arXiv:2310.19923 (2023)
6. Harley, A.W., Ufkes, A., Derpanis, K.G.: Evaluation of deep convolutional nets for document image classification and retrieval. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), pp. 991–995 (2015)
7. Hu, E.J., et al.: LoRA: low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685 (2021)
8. Jiang, A.Q., et al.: Mistral 7B. arXiv preprint arXiv:2310.06825 (2023)
9. Jiang, A.Q., et al.: Mixtral of experts. arXiv preprint arXiv:2401.04088 (2024)
10. Kim, G., et al.: OCR-free document understanding transformer. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 498–517. Springer (2022)
11. Lewis, D., Agam, G., Argamon, S., Frieder, O., Grossman, D., Heard, J.: Building a test collection for complex document information processing. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2006), pp. 665–666
12. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
13. Neelakantan, A., et al.: Text and code embeddings by contrastive pre-training. arXiv preprint arXiv:2201.10005 (2022)
14. Park, S., et al.: CORD: a consolidated receipt dataset for post-OCR parsing. In: Document Intelligence Workshop at Neural Information Processing Systems (2019)
15. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training. OpenAI (2018)
16. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)

17. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
18. Tensmeyer, C., Martinez, T.: Analysis of convolutional neural networks for document image classification. In: Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), pp. 388–393 (2017)
19. Wolf, T., et al.: HuggingFace's transformers: state-of-the-art natural language processing. arXiv preprint arXiv:1910.03771 (2019)
20. Xu, Y., et al.: LayoutLMv2: multi-modal pre-training for visually-rich document understanding. In: Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP), pp. 2579–2591 (2021)
21. Zhang, D., et al.: MM-LLMs: recent advances in multimodal large language models. arXiv preprint arXiv:2401.13601 (2024)

Towards Deployable OCR Models for Indic Languages

Minesh Mathew, Ajoy Mondal(✉), and C. V. Jawahar

CVIT, International Institute of Information Technology,
Hyderabad, India
minesh.mathew@gmail.com, {ajoy.mondal,jawahar}@iiit.ac.in

Abstract. The difficulty of reliably extracting characters had delayed the character recognition solutions (or OCRs) in Indian languages. Contemporary research in Indian language text recognition has shifted towards recognizing text in word or line images without requiring subword segmentation, leveraging Connectionist Temporal Classification (CTC) for modeling unsegmented sequences. The next challenge is the lack of public data for all these languages. And there is an immediate need to lower the entry barrier for startups or solution providers. With this in mind, (i) we introduce *Mozhi* dataset, a novel public dataset comprising over 1.2 million annotated word images (equivalent to approximately 120 thousand text line images) across 13 languages. (ii) We conduct a comprehensive empirical analysis of various neural network models employing CTC across 13 Indian languages. (iii) We also provide APIs for our OCR models and web-based applications that integrate these APIs to digitize Indic printed documents. We compare our model's performance with popular publicly available OCR tools for end-to-end document image recognition. Our model outperform these OCR engines on 8 out of 13 languages. The code, trained models, and dataset are available at https://cvit.iiit.ac.in/usodi/tdocrmil.php.

Keywords: Printed text · Indic OCR · Indian languages · CRNN · CTC · text recognition · APIs · web-based application

1 Introduction

Text recognition faces challenges related to language/script, text rendering, and imaging methods. This study concentrates on recognizing printed text in Indian languages, particularly on text recognition alone, assuming cropped word or line images are provided. The 2011 official census of India [1] lists 30 Indian languages with over a million native speakers, 22 of which are recognized as official languages. These languages belong to three language families: *Indo-European*, *Dravidian*, and *Sino-Tibetan*. Our focus is on text recognition in 13 official languages: *Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Manipuri,*

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78495-8_11.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 167–182, 2025.
https://doi.org/10.1007/978-3-031-78495-8_11

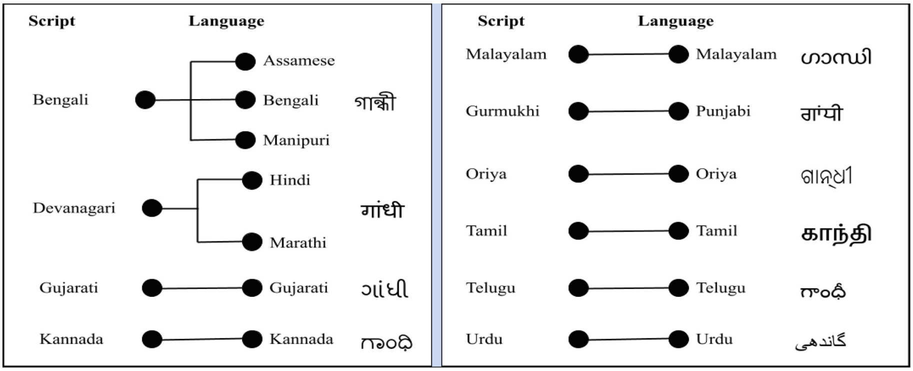


Fig. 1. We explore printed text recognition across 13 Indian languages, covering ten unique scripts. Although many languages share a common alphabet, their scripts vary, with exceptions like Hindi and Marathi. The last column shows the name "Gandhi" in all ten scripts.

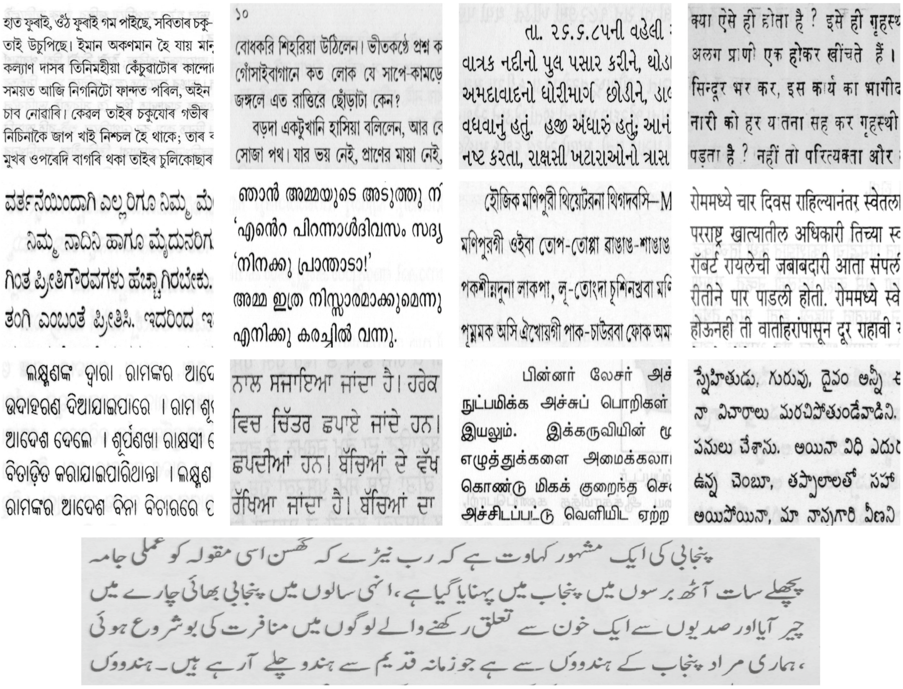


Fig. 2. Shows a few sample of cropped images of each of 13 languages from our *Mozhi* dataset.

Marathi, Oriya, Punjabi, Tamil, Telugu, and Urdu. While some share linguistic similarities, their scripts are distinct, with Devanagari script used in Hindi and Marathi and Bengali script in Bengali, Assamese, and Manipuri, among others.

Our study explores printed text recognition across 13 Indian languages, representing ten scripts. Figure 1 illustrates "Gandhi" written in these ten scripts. At the same time, Fig. 2 depicts a sample of cropped images from 13 languages from our newly created *Mozhi* dataset. The APIs corresponding to our developed models are integrated into Bhashini[1] for public use. However, we are continuously working on including the remaining low-resource languages—*Bodo, Dogri, Kashmiri, Konkani, Maithili, Nepali, Sanskrit, Santali, and Sindhi*—to cover all twenty-two languages of India.

Efforts to develop OCRs for Indian scripts began in the 1970s but faced challenges in scaling across languages and achieving satisfactory results across diverse document types until recently [4,6,29]. Challenges such as script intricacies, linguistic diversity, and limited annotated data hindered progress in Indian language OCR. The adoption of Connectionist Temporal Classification (CTC), initially successful in speech transcription, revolutionized text recognition across various forms, including handwritten [11], printed [26,31], and scene text [28,30]. Popular open-source OCR tools like *Tesseract* [2], *EasyOCR* [15], and *ocropy* [20] now leverage CTC-based models, enabling recognition of word or line images without sub-word segmentation.

Segmenting words into sub-word units presents a significant challenge for Indian languages compared to English [25]. Developing Indian language recognizers is further complicated by the intricate relationships between script glyphs, language text, and machine representation. In the script, the atomic unit is an isolated symbol (glyph), while in the language, it's an *Akshara* or an orthographic syllable. Machine text representation uses *Unicode* points. An *Akshara* can comprise multiple glyphs, and a sequence of multiple *Unicode* points can represent an *Akshara*. Splitting text at *Akshara*s and mapping them to *Unicode* sequences necessitates language and script knowledge [19,25]. Therefore, adopting CTC-based sequence modeling has become the standard approach for Indian language OCR [3,17,25]. This approach directly maps features from word or line images to target *Unicode* sequences, eliminating the need for explicit alignment during training. Our study offers a comprehensive empirical analysis of various design considerations in developing a CTC-based printed text recognition model for Indian languages.

Our contributions are the following:

- We introduce a new public dataset *Mozhi* for text recognition in 13 Indian languages, comprising cropped line and word segments with corresponding ground truth for all languages except Urdu. With over 1.2 million annotated word images, this dataset is the largest for text recognition in Indian languages (refer Table 1 and Fig. 3).
- We empirically compare the performance of four types of CTC-based text recognition methods across 13 official languages of India, varying in feature extraction and sequence encoding. Additionally, we assess word level and line level recognition models.

[1] https://bhashini.gov.in/.

- We develop end-to-end page level OCR systems by integrating our best text recognition models with existing line and word segmentation tools. These systems outperform *Tesseract5* [2] and *Google Cloud Vision OCR* [9] for 8 out of 13 languages (refer Table 4).
- Offer APIs for our OCR models and web-based applications that seamlessly integrate these APIs to digitize Indic printed documents.

2 Related Work

Current OCRs for Indian scripts mainly rely on segmentation-free approaches, which directly produce a label sequence from word or line images. Sankaran *et al.* [26] introduced CTC-based sequence modeling for printed text recognition in Indian languages. Their method utilizes an RNN encoder and CTC transcription to map features extracted from Devanagari word images to class labels. Profile-based features [32] extracted using a 25 × 1 sliding window are employed. Initially, the model maps *Aksharas* to class labels and uses rule-based mapping to Unicode. In a subsequent work [25], they directly map feature sequences from word images to *Unicode* sequences, eliminating the need for rule-based *Akshara* to *Unicode* mapping.

The introduction of the CTC-based transcription method marked a significant advancement in Indic scripts, particularly by overcoming the challenge of sub-word segmentation. Directly transcribing word images into machine-readable *Unicode* sequences also eliminated the need for language-specific rules to map latent output classes to valid *Unicode* sequences. Krishnan *et al.* [17] utilized profile-based features and a CTC-based model similar to [25] for recognizing seven Indian languages. Their evaluation on a large test set per language demonstrated the effectiveness of a unified framework employing CTC transcription for multilingual text recognition, eliminating the necessity for language or script-specific modules.

Hasan *et al.* [3] proposed an RNN+CTC model for printed Urdu text recognition, directly generating Unicode sequences from text line images. Utilizing a 30 × 1 sliding window for raw pixel feature extraction, their method yielded promising outcomes. Similarly, our prior work [19] centered on multilingual OCR for 12 Indian languages and English, employing a two-stage system with a script identification module and a recognition module. Chavan *et al.* [7] compared RNN and multidimensional RNN (MDRNN) encoders with CTC transcription. They found the MDRNN encoder outperformed the RNN encoder, using HOG features with the former and raw pixels with the latter. Another study achieved over 99% character/symbol accuracy for Bengali script recognition [22] using an RNN+CTC model. Kundaikar and Pawar [18] explored the robustness of CTC-based Devanagari OCR to font and size variations. At the same time, Dwivedi *et al.* [8] achieved a character/symbol error rate under 3% for Sanskrit recognition using an encoder-decoder model. These findings, particularly the reliance on CTC transcription, motivate our comprehensive empirical study comparing various encoder types and features for both line and word recognition in Indian languages.

3 Mozhi Dataset

To our knowledge, no extensive public datasets are available for printed text recognition in Indian languages. Early studies often utilized datasets with cropped characters or isolated symbols for character classification [5,24]. Later research relied on either internal datasets or large-scale synthetically generated samples for word or line level annotations [3,7,8,16–19,26]. While recent efforts have introduced public datasets for Hindi and Urdu, they typically contain a limited number of samples intended solely for model evaluation [16,19]. However, due to variations in training data among these studies, comparing methods can be challenging. To address the scarcity of annotated data for training printed text recognition models in Indian languages, we introduce the *Mozhi* dataset. This public dataset encompasses both line and word level annotations for all 13 languages examined in this study. It includes cropped line images, corresponding ground truth text annotations for all languages, and word images and ground truths for all languages except Urdu. With 1.2 million word annotations (approximately 100,000 words per language), it is the largest public dataset of real word images for text recognition in Indian languages. For each language, the line level data is divided randomly into training, validation, and test splits in an 80:10:10 ratio, with words cropped from line images forming corresponding splits for training, validation, and testing. Table 1 shows statistics of *Mozhi*.

Table 1. Statistics for the new *Mozhi* dataset, a public resource for recognizing printed text in cropped words and lines, reveal over 1.2 million annotated words in total. Notably, only cropped lines are annotated for Urdu.

Script	Language	Train		Validation		Test	
		Lines	Words	Lines	Words	Lines	Words
Bengali	Assamese	9566	79959	1196	9945	1196	10146
Bengali	Bengali	7579	80113	948	9787	947	10113
Gujarati	Gujarati	8632	79910	1080	10016	1079	10090
Devanagari	Hindi	6525	79762	816	10114	816	10173
Kannada	Kannada	3462	80085	1683	10088	1683	9838
Malayalam	Malayalam	15112	80146	1889	9893	1889	9980
Bengali	Manipuri	9765	79691	1221	10254	1221	10061
Devanagari	Marathi	8380	80151	1048	10005	1048	9855
Oriya	Oriya	8260	79945	1033	10089	1033	9994
Gurumukhi	Punjabi	6726	79931	841	10036	841	10038
Tamil	Tamil	16074	80022	2010	10021	2009	9974
Telugu	Telugu	12722	80337	1591	9811	1590	9876
Nastaliq	Urdu	9100	–	1138	–	1137	–

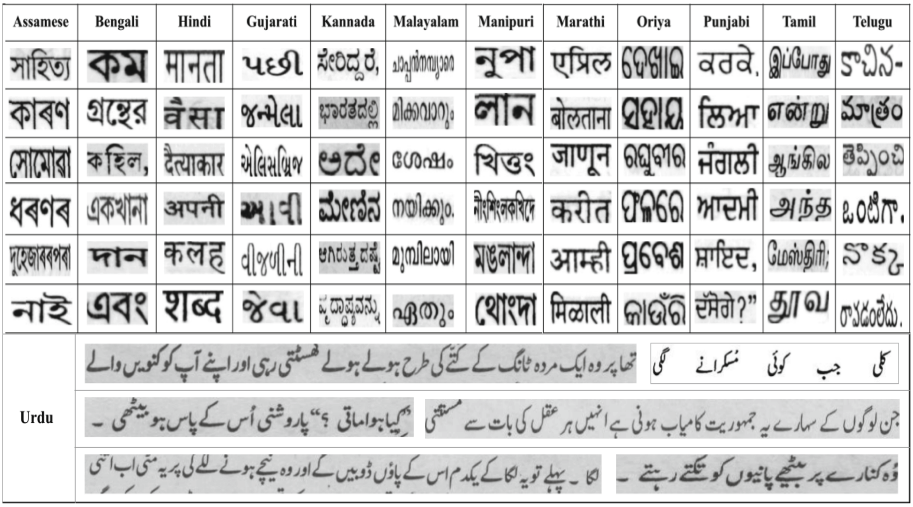
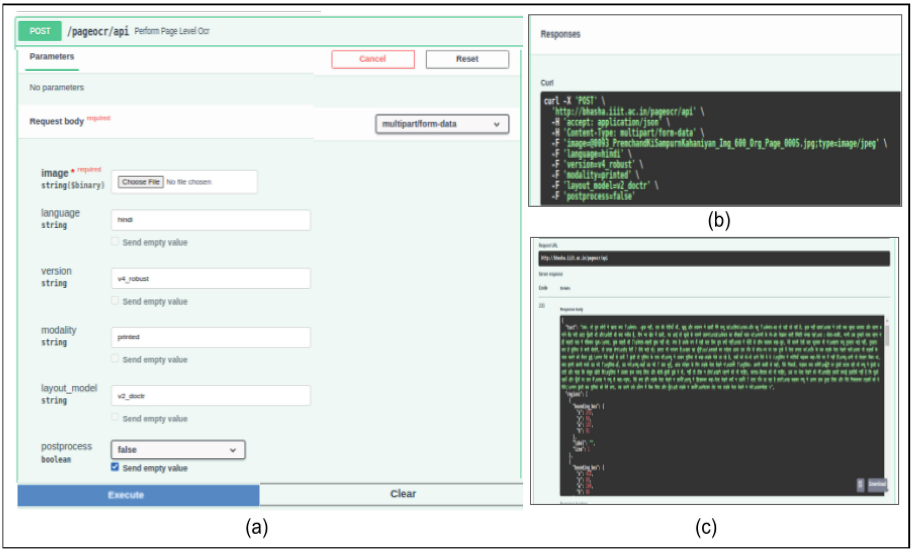

Assamese
Bengali
Hindi
Gujarati
Kannada
Malayalam
Manipuri
Marathi
Oriya
Punjabi
Tamil
Telugu
Urdu

Fig. 3. A few sample of word level images from our *Mozhi* dataset.


POST /pageocr/api Perform Page Level Ocr

Parameters

No parameters

Request body required — multipart/form-data

image * required string($binary) — Choose File No file chosen

language string — hindi
□ Send empty value

version string — v4_robust
□ Send empty value

modality string — printed
□ Send empty value

layout_model string — v2_doctr
□ Send empty value

postprocess boolean — false
☑ Send empty value

Responses

Curl

```
curl -X 'POST' \
  'http://bhasha.iiit.ac.in/pageocr/api' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'image=@0093_PremchandKiSampurnKahaniyan_Img_600_Org_Page_0005.jpg;type=image/jpeg' \
  -F 'language=hindi' \
  -F 'version=v4_robust' \
  -F 'modality=printed' \
  -F 'layout_model=v2_doctr' \
  -F 'postprocess=false'
```

Request URL

```
http://bhasha.iiit.ac.in/pageocr/api
```

Fig. 4. Shows screen shot of our web-based APIs to digitize Indic printed documents.

4 APIs and Web-Based Applications

We develop APIs for page level recognition models across 13 languages and built a web-based application available at https://ilocr.iiit.ac.in/fastocr/ that integrates these APIs for digitizing printed documents in Indic languages. Figure 4 illustrates the steps for utilizing our web-based APIs to digitize Indic printed documents. Users can upload a document image, select the language, OCR model version, layout version, and execute to obtain OCR output.

5 Text Recognition Using CTC Transcription

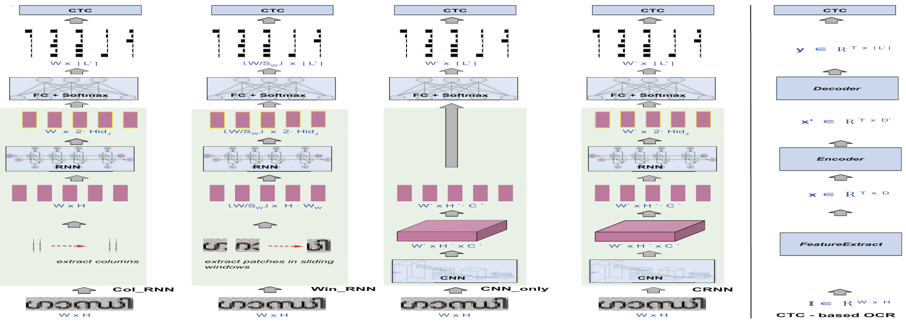


Fig. 5. We examine four CTC-based text recognition methods—Col_RNN, Win_RNN, CNN_only, and CRNN, distinguished by their feature extraction and sequence encoding. W and H represent the width and height of the input image I, respectively. $|L'|$ indicates the number of class labels, including the *blank* label. Hid_j signifies the number of hidden units in the last RNN layer. In the case of Win_RNN, W_W, and S_W denote the width and step size of the sliding window, respectively.

Given an input image I containing a word or a line, text recognition involves converting the text on the image into a machine-readable format. We frame this task as a sequence modeling problem utilizing CTC. The input comprises a sequence of features $\mathbf{x} = x_1, x_2, ..., x_T$, where $x_t \in \mathbb{R}^D$ is extracted from the image I. The output is a sequence of class labels $\mathbf{l} = l_1, l_2, ..., l_N$, where $l_n \in L$ and L represents the output alphabet, i.e., the set of unique class labels. In our scenario, L corresponds to all *Unicode* code points we aim to recognize. We adopt an encoder-decoder interpretation of the CTC framework, as described in [12] (Fig. 5).

5.1 Extracting Feature Sequence

Graves *et al.* [10] introduced CTC for speech-to-text transcription, employing a sliding window method to extract features from the time axis of the speech signal. They used a window size of 10 milliseconds (ms) and a step size of 5 ms, extracting a fixed-size feature vector termed a time-step or a frame at each instance of the sliding window. However, grey-scale images represent 2D scalar-valued spatial signals in contrast to speech signals. Thus, approaches employing CTC for text transcription from images typically extract features along the horizontal axis of the image [3,25,28]. We follow a methodology similar to that outlined in [3,25,28], where feature vectors in the input sequence $\mathbf{x}$ represent horizontal segments of the image. Each instance of the input sequence is referred to as a time-step or a frame, consistent with the original approach [10]. The horizontal span of a frame varies depending on the feature extraction method. The feature sequence, $\mathbf{x}$, is extracted in alignment with the script direction. Specifically, for languages other than Urdu, features are extracted from left to right, whereas they are extracted in the opposite direction for Urdu. In summary, given a document image $I \in \mathbb{R}^{W \times H}$ (grey-scale), the feature sequence is obtained as follows:

$$\mathbf{x} \in \mathbb{R}^{T \times D} = FeatureExtract(I). \tag{1}$$

Encoder: The sequence encoder's task is to transform the input sequence $\mathbf{x}$ into an encoded representation $\mathbf{x}' \in \mathbb{R}^{T \times D'}$, where D' represents the encoding size—i.e., the fixed dimensional to which each feature vector is encoded.

$$\mathbf{x}' \in \mathbb{R}^{T \times D'} = Encoder(\mathbf{x}). \tag{2}$$

In this work, we explore several encoder configurations—Col_RNN, Win_RNN, CNN_only, and CRNN for feature extraction[2].

Decoder: The encoded features $\mathbf{x}'$ undergo a linear projection layer followed by Softmax normalization, aligning their size with the number of output classes. This procedure, resembling the decoding phase of CTC as interpreted in [12], extends the original output alphabet L with an extra label for blank, denoted as $\sim$. The blank label signifies instances where no label is assigned to an input. Softmax normalization at each time step yields class conditional probabilities, forming the posterior distribution over the classes. Essentially, given the sequence of encoded features,

$$\mathbf{y} \in \mathbb{R}^{T \times L'} = Decoder(\mathbf{x}'), \tag{3}$$

where each $y_t \in R^{L'}$ represent activations at time step t. Thus y_t^k is a score indicating the probability of k^{th} label at time step t.

We utilize CTC transcription[3] to determine the most likely sequence of class labels given $\mathbf{y}$.

5.2 Training

Let the training dataset be denoted as $S = I_i, \mathbf{l}_i$, where I_i represents a word or line image and $\mathbf{l}_i$ represents its corresponding ground truth labeling. The objective function for training the encoder-decoder neural network for CTC transcription is derived from Maximum Likelihood principles. The aim is to minimize this objective function to maximize the log-likelihoods of the ground truth labeling. Therefore, the objective function utilized is:

$$\mathbb{O} = -\sum_{I_i, \mathbf{l}_i \in S} \log p(\mathbf{l}_i | \mathbf{y}_i), \tag{4}$$

[2] Details of them are presented in the supplementary material.

[3] Additional information regarding CTC transcription can be found in the supplementary material.

where $\mathbf{y}_i$ is the decoder output for the i^{th} sample. The above objective function can be optimized using gradient descent and back-propagation.

5.3 Inference

During inference, the CTC-based classifier aims to output the labeling $\mathbf{l}^*$ with the highest probability, as defined in Eq. (5).

$$p(\mathbf{l}|\mathbf{x}) = \sum_{\pi \in \boldsymbol{\mathcal{B}}^{-1}(\mathbf{l})} p(\pi|\mathbf{x}). \tag{5}$$

6 Experimental Setup

6.1 Implementation Details

In all experiments, cropped word or line images are resized to a height of 32 pixels and converted to grayscale, maintaining the original aspect ratio. To establish a validation split, we randomly select 5% of pages from each book in the train split for all languages. It ensures that the validation split reflects the pages in the train split while the test split comprises pages from different sets of books. In Win_RNN, the sliding window width W_W is set to 20, and the step size W_S is set to 5. For Col_RNN, Win_RNN, and CRNN, we utilize a bi-directional LSTM with 256 hidden units per direction across two layers, resulting in an output size of 2×256 at each time step. The CNN architecture in CNN_only and CRNN follows the original CRNN paper [28]. Our models are implemented using PyTorch [21]. We utilize an existing CRNN implementation [14] for our experiments, conducting training on a single Nvidia GeForce 1080 Ti GPU. Training is set for 30 epochs. Word recognition models have a batch size of 64, while line recognition models use a batch size of 16. RMSProp [13] is employed as the optimizer. Col_RNN and Win_RNN are assigned a learning rate of $10e-03$, while CNN_only and CRNN variants converge faster with a lower learning rate of $10e-04$.

6.2 Evaluation

We need to assess text recognition in three scenarios: (i) word OCR: recognizing cropped word images, (ii) line OCR: recognizing cropped line images, and (iii) page OCR: end-to-end text recognition from document images. Our main evaluation metric in all cases is Character Accuracy (CA), determined by the Levenshtein distance between predicted and ground truth strings. For a formal definition of CA, let us denote the predicted text for a word/line/page as l_i and the corresponding ground truth as g_i. If there are N such samples, CA is defined as

$$CA = \frac{\sum_i len(g_i) - \sum_i LD(l_i, g_i)}{\sum_i len(g_i)} \times 100, \tag{6}$$

where len is a function that returns the length of the given string, and LD is a function that computes the Levenshtein distance between the given pair of strings. Note that Character Error Rate (CER), another commonly used metric for OCR evaluation, is essentially $100 - CA$. We also include Sequence Accuracy (SA) alongside CA for word OCR and line OCR. SA represents the percentage of samples where the prediction is entirely correct (i.e., $LD(l_i, g_i) = 0$). In the context of word recognition models, SA is equivalent to 'word accuracy' and is commonly used in scene text recognition literature.

Table 2. Results for recognition-only tasks are presented for each language individually on validation set of *Mozhi* dataset. Each model configuration (Col_RNN, Win_RNN, CNN_only, and CRNN) is trained separately for each language. Character Accuracy (CA) and Sequence Accuracy (SA) are reported for word recognition. The highest CA and SA values among the four encoder configurations are highlighted in bold.

Language	Word Recognition							
	Col_RNN		Win_RNN		CNN_only		CRNN	
	CA	SA	CA	SA	CA	SA	CA	SA
Assamese	98.6	95.4	97.6	92.9	98.3	96.0	**99.0**	**96.5**
Bengali	99.1	97.0	98.3	94.5	99.2	97.3	**99.4**	**97.9**
Guajrati	96.2	92.4	95.1	89.5	96.2	90.9	**96.5**	**93.9**
Hindi	97.6	95.1	96.3	92.3	97.4	94.2	**98.2**	**96.3**
Kannada	97.4	88.9	96.4	84.7	96.7	85.8	**97.7**	**90.7**
Malayalam	99.5	96.6	99.3	95.6	98.0	83.7	**99.7**	**97.7**
Manipuri	98.6	95.4	97.8	92.8	98.2	93.1	**99.0**	**96.9**
Marathi	99.0	96.2	98.5	94.2	98.9	95.0	**99.2**	**96.9**
Odia	96.8	93.5	95.7	90.8	96.9	93.7	**97.2**	**94.8**
Punjabi	99.1	97.7	98.4	96.4	99.2	97.8	**99.5**	**98.7**
Tamil	97.9	91.0	97.4	88.4	97.3	87.2	**98.0**	**91.8**
Telugu	96.3	91.4	95.3	86.8	96.4	92.0	**96.8**	**93.6**
Urdu	–	–	–	–	–	–	–	–

We employ a standard OCR evaluation toolkit for page OCR, where the input is a document image. Specifically, we utilize a modern adaptation [27] of the original ISRI Analytic Tools for OCR Evaluation [23]. Using this toolkit, we compute Character Accuracy (CA) and Word Accuracy (WA). CA is calculated following the method described in Eq. (6). Word accuracy is determined by aligning the sequences of words in the prediction l_i with those in the ground truth g_i and identifying the Longest Common Sub-sequence (LCS) between them. For a set of pages,

$$WA = \frac{\sum_i len(LCS(l_i, g_i))}{\sum_i len(g_i)} \times 100 \tag{7}$$

where *len* returns the number of words in a given sequence of words.

7 Experiments and Results[4]

7.1 Comparing Different Encoder Configurations

We assess the performance of four encoder configurations on the validation set of *Mozhi* dataset for word recognition. Results are presented in Table 2. Each CA and SA pair in the table corresponds to a CTC-based network trained separately for a specific combination of language, recognition unit (word), and encoder configuration (Col_RNN, Win_RNN, CNN_only, and CRNN). Across all cases except for Urdu word recognition, CRNN emerges as the top performer among the four configurations. The superior performance of CRNN over the CNN configuration highlights the necessity of capturing long-term dependencies in word or line images. Unlike fully connected networks, CNN layers have limited receptive fields, necessitating numerous layers to cover the entire input. Our seven-layer CNN lacks the depth to model extensive horizontal dependencies adequately. This deficiency is mitigated by employing a sequence encoder (bi-directional LSTM) that proficiently captures long-term dependencies in both directions.

Table 3. CRNN evaluation on test set of *Mozhi* dataset. For each language, we train both word and line level CRNN models on the respective train split of the *Mozhi* dataset.

Language	Test			
	Word		Line	
	CA	SA	CA	SA
Assamese	98.9	96.2	99.2	76.8
Bengali	99.0	96.9	98.1	68.4
Gujarati	98.0	94.9	97.4	63.1
Hindi	98.1	95.5	98.8	63.5
Kannada	97.1	88.7	97.5	53.9
Malayalam	99.5	97.3	99.5	87.3
Manipuri	98.4	95.9	99.2	79.4
Marathi	99.3	97.0	99.3	73.8
Oriya	97.5	94.3	98.8	73.1
Punjabi	99.2	98.2	99.3	79.7
Tamil	98.0	91.6	98.3	68.1
Telugu	99.1	95.4	98.9	71.7
Urdu	–	–	93.8	24.2

7.2 Evaluating CRNN on Test Set of *Mozhi*

Table 2 highlights that among four different models—Col_RNN, Win_RNN, CNN_only, and RCNN, RCNN obtained the best results for all languages on

[4] Additional results can be found in the supplementary material.

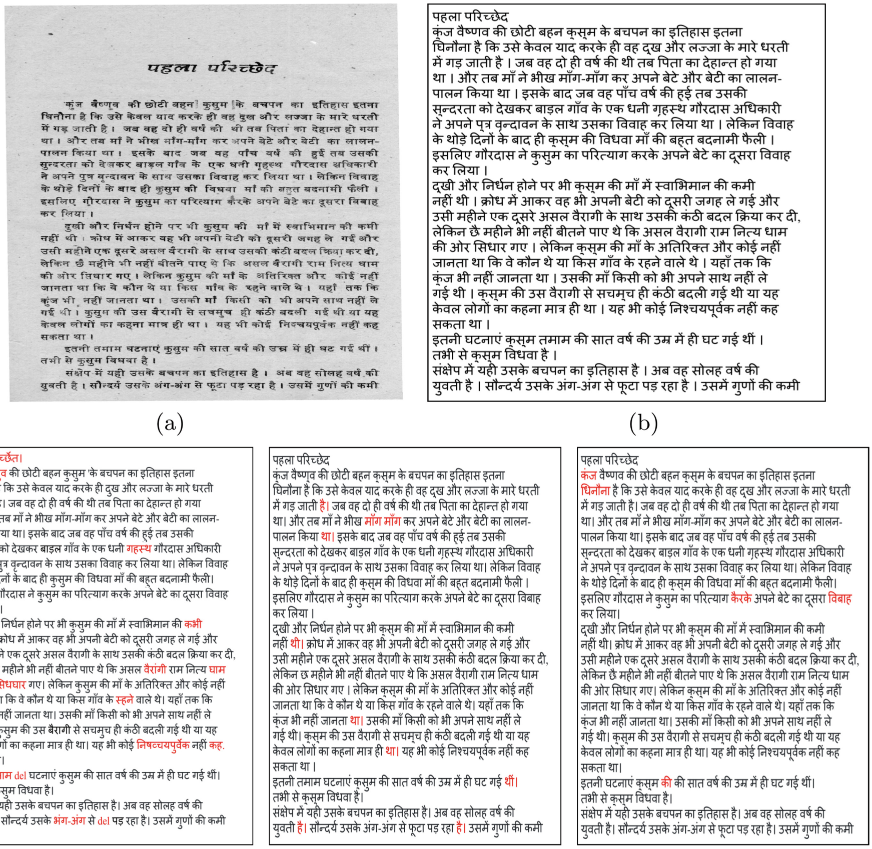
Table 4. Performance of our page OCR pipelines compared to other public OCR tools. In this setting, we evaluate text recognition in an end-to-end manner on the test split of our dataset. Since the focus of this work is on text recognition, for end-to-end settings, for text detection, gold standard word/line bounding boxes are used. Under 'End-to-End OCR' we show results of *Tesseract* [2] and *Google Cloud Vision OCR* [9]. Given a document image, these tools output a transcription of the page along with the bounding boxes of the lines and words detected. Under 'GT Detection+CRNN', we show results of an end-to-end pipeline where gold standard word and line detection are used. For instance, 'GT Word' means we used ground truth (GT) word bounding boxes and the CRNN model trained for recognizing words, for that particular language. Bold value indicates the best result.

Language	End-to-End OCR				GT Detection+CRNN			
	Tesseract		Google		GT word		GT line	
	CA	SA	CA	SA	CA	SA	CA	SA
Assamese	92.7	91.2	90.0	86.0	99.3	97.0	**99.4**	**97.2**
Bengali	93.5	96.2	84.0	91.3	**99.1**	**97.3**	99.0	96.8
Gujarati	96.9	92.4	93.0	**95.2**	**98.0**	93.7	97.7	91.9
Hindi	95.0	93.3	95.2	**97.3**	**98.1**	96.0	98.0	95.6
Kannada	94.9	85.1	85.7	84.6	95.6	**89.2**	**95.9**	86.4
Malayalam	96.2	78.7	88.0	74.8	**99.4**	**98.0**	99.3	97.9
Manipuri	90.9	80.6	85.7	77.4	98.4	94.7	**98.7**	**94.9**
Marathi	97.9	97.4	98.3	**98.4**	**99.6**	98.2	99.5	98.0
Oriya	94.0	83.6	92.6	90.0	**98.6**	**95.4**	98.0	94.5
Punjabi	93.2	89.8	92.7	96.7	99.2	**98.3**	**99.3**	97.9
Tamil	79.3	42.4	92.5	**93.1**	96.1	85.6	*96.5*	85.4
Telugu	93.7	79.3	94.2	89.2	**99.1**	**95.1**	98.9	94.0
Urdu	68.3	26.2	92.7	**85.7**	–	–	**94.7**	81.5

validation set of *Mozhi* dataset with respect to CA and SA metrices for word recognition task. Since RCNN, highest performing model for validation set, we evaluated these models on test set of the same dataset. Table 3 presents obtained results for word and line recognition on test set.

7.3 Page Level OCR Evaluation

In page level OCR, the goal is to transcribe the text within a document image by segmenting it into lines or words and then recognizing the text at the word or line level. Our focus lies solely on text recognition, excluding layout analysis and reading order identification. To construct an end-to-end page OCR pipeline, we combine existing text detection methods with our CRNN models for recognition. Transcriptions from individual segments are arranged in the detected reading order. We evaluate the end-to-end pipeline by using gold standard detection to establish an upper bound on our CRNN model's performance. Additionally, we compare our OCR results with two public OCR tools: *Tesseract* and *Google Cloud Vision OCR*. Results from all end-to-end evaluations are summarized in Table 4.



(a)

पहला परिच्छेद
कुंज वैष्णव की छोटी बहन कुसुम के बचपन का इतिहास इतना
घिनौना है कि उसे केवल याद करके ही वह दुख और लज्जा के मारे धरती
में गड़ जाती है । जब वह दो ही वर्ष की थी तब पिता का देहान्त हो गया
था । और तब माँ ने भीख माँग-माँग कर अपने बेटे और बेटी का लालन-
पालन किया था । इसके बाद जब वह पाँच वर्ष की हुई तब उसकी
सुन्दरता को देखकर बाड़ल गाँव के एक धनी गृहस्थ गौरदास अधिकारी
ने अपने पुत्र वृन्दावन के साथ उसका विवाह कर लिया था । लेकिन विवाह
के थोड़े दिनों के बाद ही कुसुम की विधवा माँ की बहुत बदनामी फैली ।
इसलिए गौरदास ने कुसुम का परित्याग करके अपने बेटे का दूसरा विवाह
कर लिया ।
दुखी और निर्धन होने पर भी कुसुम की माँ में स्वाभिमान की कमी
नहीं थी । क्रोध में आकर वह भी अपनी बेटी को दूसरी जगह ले गई और
उसी महीने एक दूसरे असल वैरागी के साथ उसकी कंठी बदल क्रिया कर दी,
लेकिन छै महीने भी नहीं बीतने पाए थे कि असल वैरागी राम नित्य धाम
की ओर सिधार गए । लेकिन कुसुम की माँ के अतिरिक्त और कोई नहीं
जानता था कि वे कौन थे या किस गाँव के रहने वाले थे । यहाँ तक कि
कुंज भी नहीं जानता था । उसकी माँ किसी को भी अपने साथ नहीं ले
गई थी । कुसुम की उस वैरागी से सचमुच ही कंठी बदली गई थी या यह
केवल लोगों का कहना मात्र ही था । यह भी कोई निश्चयपूर्वक नहीं कह
सकता था ।
इतनी घटनाएं कुसुम तमाम की सात वर्ष की उम्र में ही घट गई थीं ।
तभी से कुसुम विधवा है ।
संक्षेप में यही उसके बचपन का इतिहास है । अब वह सोलह वर्ष की
युवती है । सौन्दर्य उसके अंग-अंग से फूटा पड़ रहा है । उसमें गुणों की कमी

(b)

पहला परर्च्छेत।
कंज वेष्णुव की छोटी बहन कुसुम 'के बचपन का इतिहास इतना
घिनौना है कि उसे केवल याद करके ही दुख और लज्जा के मारे धरती
में जाती है। जब वह दो ही वर्ष की थी तब पिता का देहान्त हो गया
था। और तब माँ ने भीख माँग-माँग कर अपने बेटे और बेटी का लालन-
पालन किया था। इसके बाद जब वह पाँच वर्ष की हुई तब उसकी
सुन्दरता को देखकर बाड़ल गाँव के एक धनी गहस्थ गौरदास अधिकारी
ने अपने पुत्र वृन्दावन के साथ उसका विवाह कर लिया था। लेकिन विवाह
के थोड़े दिनों के बाद ही कुसुम की विधवा माँ की बहुत बदनामी फैली।
इसलिए गौरदास ने कुसुम का परित्याग करके अपने बेटे का दूसरा विवाह
कर लिया।
दुखी और निर्धन होने पर भी कुसुम की माँ में स्वाभिमान की कभी
नहीं थी। क्रोध में आकर वह भी अपनी बेटी को दूसरी जगह ले गई और
उसी महीने एक दूसरे असल वैरागी के साथ उसकी कंठी बदल क्रिया कर दी,
लेकिन छे महीने भी नहीं बीतने पाए थे कि असल वैरांगी राम नित्य घाम
की ओर सिधघार गए। लेकिन कुसुम की माँ के अतिरिक्त और कोई नहीं
जानता था कि वे कौन थे या किस गाँव के स्हने वाले थे। यहाँ तक कि
कुंज भी. नहीं जानता था। उसकी माँ किसी को भी अपने साथ नहीं ले
गई थी। कुसुम की उस बैरागी से सचमुच ही कंठी बदली गई थी या यह
केवल लोगों का कहना मात्र ही था। यह भी कोई निषव्चयपुर्वेक नहीं कह.
सकता था।
इतनी तमाम del घटनाएं कुसुम की सात वर्ष की उम्र में ही घट गई थीं।
तभी से कुसुम विधवा है।
संक्षेप में यही उसके बचपन का इतिहास है। अब वह सोलह वर्ष की
युवती है। सौन्दर्य उसके भंग-अंग से del पड़ रहा है। उसमें गुणों की कमी

(c)

पहला परिच्छेद
कुंज वैष्णव की छोटी बहन कुसुम के बचपन का इतिहास इतना
घिनौना है कि उसे केवल याद करके ही वह दुख और लज्जा के मारे धरती
में गड़ जाती है। जब वह दो ही वर्ष की थी तब पिता का देहान्त हो गया
था। और तब माँ ने भीख माँग माँग कर अपने बेटे और बेटी का लालन-
पालन किया था। इसके बाद जब वह पाँच वर्ष की हुई तब उसकी
सुन्दरता को देखकर बाड़ल गाँव के एक धनी गृहस्थ गौरदास अधिकारी
ने अपने पुत्र वृन्दावन के साथ उसका विवाह कर लिया था। लेकिन विवाह
के थोड़े दिनों के बाद ही कुसुम की विधवा माँ की बहुत बदनामी फैली ।
इसलिए गौरदास ने कुसुम का परित्याग करके अपने बेटे का दूसरा विबाह
कर लिया ।
दुखी और निर्धन होने पर भी कुसुम की माँ में स्वाभिमान की कमी
नहीं थी। क्रोध में आकर वह भी अपनी बेटी को दूसरी जगह ले गई और
उसी महीने एक दूसरे असल वैरागी के साथ उसकी कंठी बदल क्रिया कर दी,
लेकिन छ महीने भी नहीं बीतने पाए थे कि असल वैरागी राम नित्य धाम
की ओर सिधार गए । लेकिन कुसुम की माँ के अतिरिक्त और कोई नहीं
जानता था कि वे कौन थे या किस गाँव के रहने वाले थे। यहाँ तक कि
कुंज भी नहीं जानता था। उसकी माँ किसी को भी अपने साथ नहीं ले
गई थी। कुसुम की उस वैरागी से सचमुच ही कंठी बदली गई थी या यह
केवल लोगों का कहना मात्र ही था। यह भी कोई निश्चयपूर्वक नहीं कह
सकता था ।
इतनी तमाम घटनाएं कुसुम की सात वर्ष की उम्र में ही घट गई थीं।
तभी से कुसुम विधवा है।
संक्षेप में यही उसके बचपन का इतिहास है। अब वह सोलह वर्ष की
युवती है। सौन्दर्य उसके अंग-अंग से फूटा पड़ रहा है। उसमें गुणों की कमी

(d)

पहला परिच्छेद
कंज वैष्णव की छोटी बहन कुसुम के बचपन का इतिहास इतना
धिनौना है कि उसे केवल याद करके ही वह दुख और लज्जा के मारे धरती
में गड़ जाती है। जब वह दो ही वर्ष की थी तब पिता का देहान्त हो गया
था। और तब माँ ने भीख माँग-माँग कर अपने बेटे और बेटी का लालन-
पालन किया था। इसके बाद जब वह पाँच वर्ष की हुई तब उसकी
सुन्दरता को देखकर बाड़ल गाँव के एक धनी गृहस्थ गौरदास अधिकारी
ने अपने पुत्र वृन्दावन के साथ उसका विवाह कर लिया था। लेकिन विवाह
के थोड़े दिनों के बाद ही कुसुम की विधवा माँ की बहुत बदनामी फैली।
इसलिए गौरदास ने कुसुम का परित्याग कैरके अपने बेटे का दूसरा विबाह
कर लिया।
दुखी और निर्धन होने पर भी कुसुम की माँ में स्वाभिमान की कमी
नहीं थी। क्रोध में आकर वह भी अपनी बेटी को दूसरी जगह ले गई और
उसी महीने एक दूसरे असल वैरागी के साथ उसकी कंठी बदल क्रिया कर दी,
लेकिन छै महीने भी नहीं बीतने पाए थे कि असल वैरागी राम नित्य धाम
की ओर सिधार गए। लेकिन कुसुम की माँ के अतिरिक्त और कोई नहीं
जानता था कि वे कौन थे या किस गाँव के रहने वाले थे। यहाँ तक कि
कुंज भी नहीं जानता था। उसकी माँ किसी को भी अपने साथ नहीं ले
गई थी। कुसुम की उस वैरागी से सचमुच ही कंठी बदली गई थी या यह
केवल लोगों का कहना मात्र ही था। यह भी कोई निश्चयपूर्वक नहीं कह
सकता था।
इतनी घटनाएं कुसुम की की सात वर्ष की उम्र में ही घट गई थीं।
तभी से कुसुम विधवा है।
संक्षेप में यही उसके बचपन का इतिहास है। अब वह सोलह वर्ष की
युवती है। सौन्दर्य उसके अंग-अंग से फूटा पड़ रहा है। उसमें गुणों की कमी

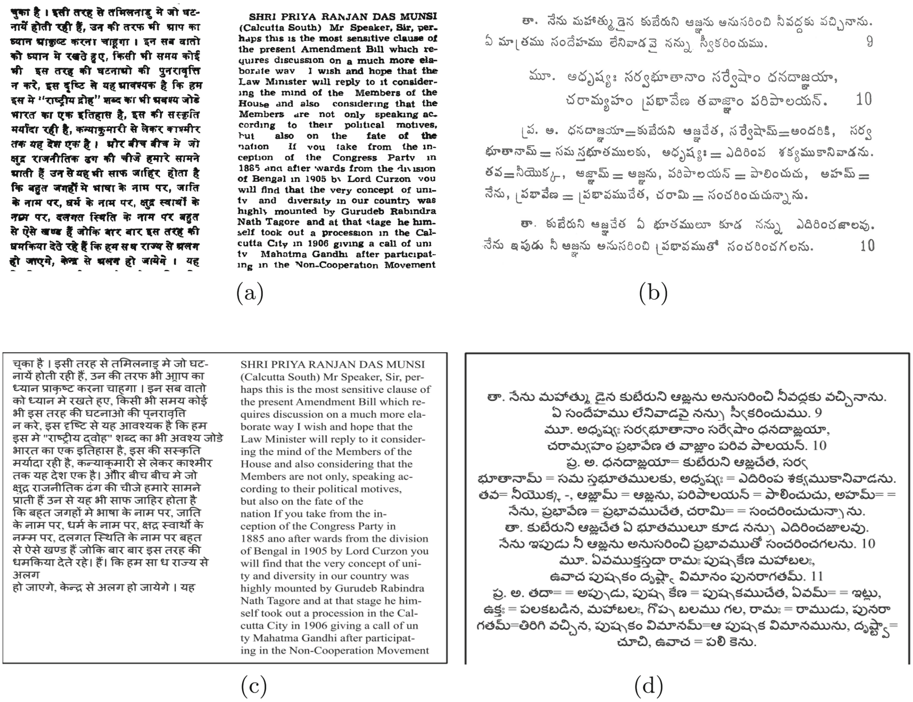
(e)

Fig. 6. Displays qualitative results at the page level using *Tesseract*, *Google OCR*, and our method on a Hindi document image. For optimal viewing, zoom in. (a) original document image, (b) ground truth textual transcription, (c) predicted text by *Tesseract*, (d) predicted text by *Google OCR*, and (e) predicted text by our approach.

In Fig. 6, visual results at the page level using *Tesseract*, *Google OCR*, and our approach are depicted. Panel (a) presents the original document image, while panels (b) to (e) display the ground truth and the predicted text by *Tesseract*, *Google OCR*, and our approach, respectively. Wrongly recognized texts are highlighted in red. This figure emphasizes that our approach outperforms existing OCR tools in producing accurate text outputs.

7.4 Use Cases

We leverage our OCR APIs for various significant applications. Notable examples include the pages of the *Punjab Vidhan Sabha*, *Loksabha records*, and *Telugu Upanishads.* These digitization efforts enable easier access, preservation, and analysis of these valuable texts. The output and effectiveness of our OCR technology in these diverse use cases are illustrated in Fig. 7. These applications showcase the versatility and reliability of our OCR APIs in handling different scripts and document types, ensuring high accuracy and efficiency.



चुका है । इसी तरह से तमिलनाडु मे जो घट-
नायें होती रही हैं, उन की तरफ भी आप का
ध्यान आकृष्ट करना चाहूगा । इन सब वातो
को ध्यान मे रखते हुए, किसी भी समय कोई
भी इस तरह की घटनाओ की पुनरावृत्ति
न करे, इस दृष्टि से यह आवश्यक है कि हम
इस मे "राष्ट्रीय द्रोह" शब्द का भी अवश्य जोडे
भारत का एक इतिहास है, इस की सस्कृति
मर्यादा रही है, कन्याकुमारी से लेकर काश्मीर
तक यह देश एक है । और बीच बीच मे जो
क्षुद्र राजनीतिक ढग की चीजे हमारे सामने
आती हैं उन से यह भी साफ जाहिर होता है
कि बहुत जगहों मे भाषा के नाम पर, जाति
के नाम पर, धर्म के नाम पर, क्षुद्र स्वार्थों के
नाम पर, दलगत स्थिति के नाम पर बहुत
से ऐसे खण्ड हैं जोकि बार बार इस तरह की
धमकिया देते रहे हैं कि हम सब राज्य से अलग
हो जाएगे, केन्द्र से अलग हो जायेगे । यह

SHRI PRIYA RANJAN DAS MUNSI
(Calcutta South) Mr Speaker, Sir, perhaps this is the most sensitive clause of the present Amendment Bill which requires discussion on a much more elaborate wav I wish and hope that the Law Minister will reply to it considering the mind of the Members of the House and also considering that the Members are not only speaking according to their political motives, hut also on the fate of the nation If vou take from the inception of the Congress Partv in 1885 and after wards from the division of Bengal in 1905 bv Lord Curzon vou will find that the very concept of unitv and diversity in our countrv was highlv mounted by Gurudeb Rabindra Nath Tagore and at that stage he himself took out a procession in the Calcutta City in 1906 giving a call of uni tv Mahatma Gandhi after participating in the Non-Cooperation Movement

(a)

తా. నేను మహాత్ముడైన కుబేరుని ఆజ్ఞను అనుసరించి నీవద్దకు వచ్చినాను.
ఏ మాత్రము సందేహము లేనివాడవై నన్ను స్వీకరించుము. 9

మూ. అధృష్యః సర్వభూతానాం సర్వేషాం ధనదాజ్ఞయా,
చరామ్యహం ప్రభావేణ తవాజ్ఞాం పరిపాలయన్. 10

ప్ర. అ. ధనదాజ్ఞయా=కుబేరుని ఆజ్ఞచేత, సర్వేషామ్=అందరికి, సర్వ
భూతానామ్=సమస్తభూతములకు, అధృష్యః=ఎదిరింప శక్యముకానివాడను.
తవ=నీయొక్క, ఆజ్ఞామ్=ఆజ్ఞను, పరిపాలయన్=పాలించుచు, అహమ్=
నేను, ప్రభావేణ=ప్రభావముచేత, చరామి=సంచరించుచున్నాను.

తా. కుబేరుని ఆజ్ఞచేత ఏ భూతములూ కూడ నన్ను ఎదిరించజాలవు.
నేను ఇపుడు నీ ఆజ్ఞను అనుసరించి ప్రభావముతో సంచరించగలను. 10

(b)

चुका है । इसी तरह से तमिलनाड़ मे जो घट-
नायें होती रही हैं, उन की तरफ भी आ़ाप का
ध्यान प्राकृष्ट करना चाहूगा । इन सब वातो
को ध्यान मे रखते हुए, किसी भी समय कोई
भी इस तरह की घटनाओ की पूनरावृत्ति
न करे, इस दृष्टि से यह आवश्यक है कि हम
इस मे "राष्ट्रीय द्वोह" शब्द का भी अवश्य जोडे
भारत का एक इतिहास है, इस की सस्कृति
मर्यादा रही है, कन्याक्मारी से लेकर काश्मीर
तक यह देश एक है। औौर बीच बीच मे जो
क्षुद्र राजनीतिक ढंग की चीजे हमारे सामने
प्राती हैं उन से यह भी साफ जाहिर होता है
कि बहृत जगहों मे भाषा के नाम पर, जाति
के नाम पर, धर्म के नाम पर, क्षद्र स्वार्थो के
नम्म पर, दलगत स्थिति के नाम पर बहृत
से ऐसे खण्ड हैं जोकि बार बार इस तरह की
धमकिया देते रहे। हैं। कि हम सा ध राज्य से
अलग
हो जाएगे, केन्द्र से अलग हो जायेगे । यह

SHRI PRIYA RANJAN DAS MUNSI
(Calcutta South) Mr Speaker, Sir, perhaps this is the most sensitive clause of the present Amendment Bill which requires discussion on a much more elaborate way I wish and hope that the Law Minister will reply to it considering the mind of the Members of the House and also considering that the Members are not only, speaking according to their political motives,
but also on the fate of the
nation If you take from the inception of the Congress Party in 1885 ano after wards from the division of Bengal in 1905 by Lord Curzon you will find that the very concept of unity and diversity in our country was highly mounted by Gurudeb Rabindra Nath Tagore and at that stage he himself took out a procession in the Calcutta City in 1906 giving a call of un ty Mahatma Gandhi after participating in the Non-Cooperation Movement

(c)

తా. నేను మహాత్ము డైన కుబేరుని ఆజ్ఞను అనుసరించి నీవద్దకు వచ్చినాను.
ఏ సందేహము లేనివాడవై నన్ను స్వీకరించుము. 9
మూ. అధృష్యః సర్వభూతానాం సర్వేషాం ధనదాజ్ఞయా,
చరామ్యహం ప్రభావేణ త వాజ్ఞాం పరివ పాలయన్. 10
ప్ర. అ. ధనదాజ్ఞయా= కుబేరుని ఆజ్ఞచేత, సర్వ
భూతానామ్ = సమ స్తభూతములకు, అధృష్యః = ఎదిరింప శక్యముకానివాడను.
తవ= నీయొక్క -, ఆజ్ఞామ్ = ఆజ్ఞను, పరిపాలయన్ = పాలించుచు, అహమ్= =
నేను, ప్రభావేణ = ప్రభావముచేత, చరామి= = సంచరించుచున్నాను.
తా. కుబేరుని ఆజ్ఞచేత ఏ భూతములూ కూడ నన్ను ఎదిరించజాలవు.
నేను ఇపుడు నీ ఆజ్ఞను అనుసరించి ప్రభావముతో సంచరించగలను. 10
మూ. ఏవముక్తస్తదా రామః పుష్పకేణ మహాబలః,
ఉవాచ పుష్పకం దృష్ట్వా విమానం పునరాగతమ్. 11
ప్ర. అ. తదా= = అప్పుడు, పుష్ప కేణ = పుష్పకముచేత, ఏవమ్= = ఇట్లు,
ఉక్తః = పలకబడిన, మహాబలః, గొప్ప బలము గల, రామః = రాముడు, పునరా
గతమ్=తిరిగి వచ్చిన, పుష్పకం విమానమ్=ఆ పుష్పక విమానమును, దృష్ట్వా=
చూచి, ఉవాచ = పలి కెను.

(d)

Fig. 7. Illustrates use cases for the digitization of *Loksabha records* and *Telugu Upanishad* pages. (a) and (b) display cropped regions from the original images of *Loksabha* and *Upanishad* documents, respectively. Panels (c) and (d) present the corresponding text outputs generated using our OCR APIs.

7.5 Discussion

Our method performs better in page level recognition than *Tesseract* across all 13 languages, as evidenced by the results in Table 4. Specifically, our approach surpasses *Google* for eight languages, as indicated in the same table when considering ground truth bounding boxes. However, our dataset predominantly comprises pages from books, resulting in limited font, style, layout, and distortion diversity. Nevertheless, this dataset can serve as valuable pre-training data. Moving forward, we aim to enrich the dataset by gathering diverse documents with varying layouts, content, fonts, styles, and distortions, enhancing its comprehensiveness and utility for developing robust recognition models.

8 Conclusions

We empirically study different CTC-based word and line recognition models in 13 Indian languages. Our study concludes that CRNN, which uses a CNN for feature representation and a dedicated RNN-based sequential encoder, works best. Using existing text detection tools and our recognition models, we build page level OCR pipeline and show that our approach works better than two popular OCR tools for most of the languages. We also introduce a new public *Mozhi* dataset for cropped word/line recognition in 13 Indian languages with more than 1.2 million annotated words. Additionally, we provide APIs for our page level OCR models and web-based applications that integrate these APIs to digitize Indic printed documents. We believe our study, the *Mozhi* dataset, and available APIs will encourage research on OCR of Indian languages.

Acknowledgment. This work is supported by MeitY, Government of India, through the NLTM-Bhashini project.

References

1. Census 2011. https://censusindia.gov.in/2011-Common/CensusData2011.Html
2. Tesseract (2021). https://github.com/tesseract-ocr/tesseract. Accessed 20 November 2021
3. Ul-Hasan, A., Ahmed, S.B., Rashid, S.F., Shafait, F., Breuel, T.M.: Offline printed Urdu Nastaleeq script recognition with bidirectional LSTM networks. In: ICDAR (2013)
4. Arya, D., et al.: Experiences of integration and performance testing of multilingual OCR for printed Indian scripts. In: J-MOCR Workshop, ICDAR (2011)
5. Jawahar, C.V., Kumar, M.P., Ravikiran, S.S.: A Bilingual OCR system for Hindi-Telugu Documents and its Applications. In: International Conference on Document Analysis and Recognition (ICDAR) (2003)
6. Chaudhuri, B.B., Pal, U.: A complete printed Bangla OCR system. Pattern Recogn. **31**, 531–549 (1998)
7. Chavan, V., Malage, A., Mehrotra, K., Gupta, M.K.: Printed text recognition using BLSTM and MDLSTM for Indian languages. In: ICIIP (2017)
8. Dwivedi, A., Saluja, R., Sarvadevabhatla, R.K.: An OCR for classical Indic documents containing arbitrarily long words. In: CVPR Workshops (2020)
9. Google: Google Cloud Vision OCR (2021). https://cloud.google.com/vision/docs/ocr. Accessed 10 Nov 2021
10. Graves, A., Fernández, S., Gomez, F.J., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: ICML (2006)

11. Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., Schmidhuber, J.: A novel connectionist system for unconstrained handwriting recognition. IEEE Trans. Pattern Anal. Mach. Intell (2009)
12. Hannun, A.: Sequence modeling with CTC. Distill (2017). https://doi.org/10.23915/distill.00008, https://distill.pub/2017/ctc
13. Hinton: Neural Networks for Machine Learning, Lecture 6 (2012). http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf. Accessed 10 Nov 2021
14. Holmeyoung: CRNN-pytorch (2019). https://github.com/Holmeyoung/crnn-pytorch. Accessed 3 Feb 2021
15. jaidedAI: Easyocr (2022). https://github.com/JaidedAI/EasyOCR
16. Jain, M., Mathew, M., Jawahar, C.V.: Unconstrained OCR for Urdu using deep CNN-RNN hybrid networks. In: ACPR, p. 6 (2017)
17. Krishnan, P., Sankaran, N., Singh, A.K., Jawahar, C.V.: Towards a robust OCR system for Indic scripts. In: DAS (2014)
18. Kundaikar, T., Pawar, J.D.: Multi-font Devanagari text recognition using LSTM neural networks. In: ICCIGST (2020)
19. Mathew, M., Singh, A.K., Jawahar, C.V.: Multilingual OCR for Indic scripts. In: DAS (2016)
20. OCRopus: OCRopy (2022). https://github.com/ocropus/ocropy
21. Paszke, A., et al.: Pytorch: an imperative style, high-performance deep learning library. In: NeurIPS (2019)
22. Paul, D., Chaudhuri, B.B.: A BLSTM network for printed Bengali OCR system with high accuracy. ArXiv abs/1908.08674 (2019)
23. Rice, S.V., Nartker, T.A.: The ISRI analytic tools for OCR evaluation version 5.1 (1996)
24. Sanjeev Kunte, R., Sudhaker Samuel, R.: A simple and efficient optical character recognition system for basic symbols in printed Kannada text. Sadhana **32**(5) (2007)
25. Sankaran, N., Jawahar, C.V.: Devanagari text recognition: a transcription based formulation. In: ICDAR (2013)
26. Sankaran, N., Jawahar, C.: Recognition of printed Devanagari text using BLSTM neural network. In: ICPR (2012)
27. Santos, E.A.: OCR evaluation tools for the 21st century. In: Workshop on the Use of Computational Methods in the Study of Endangered Languages (2019)
28. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. CoRR (2015)
29. Sinha, R.M.K., Mahabala, H.: Machine recognition of Devanagari script. In: IEEE Trans. SMC (1979)
30. Su, B., Lu, S.: Accurate scene text recognition based on recurrent neural network. In: ACCV (2014)
31. Breuel, T.M., Ul-Hasan, A., Al Azawi, M.I.A., Shafait, F.: High-performance OCR for printed english and fraktur using LSTM networks. In: ICDAR (2013)
32. Rath, T.M., Manmatha, R.: Features for word spotting in historical manuscripts. In: ICDAR (2003)

XLSI: A New Xception and Log Polar Transform Based Approach for Scene Text Script Identification

Ayush Roy[1], Shivakumara Palaiahnakote[2](✉), Umapada Pal[1],
Apostolos Antonacopoulos[2], and Michael Blumenstein[3]

[1] Computer Vision and Pattern Recognition, Indian Statistical Institute, Kolkata, Kolkata, India
umapada@isical.ac.in

[2] School of Science, Engineering and Environment, University of Salford, Manchester, UK
s.palaiahnakote@salford.ac.uk,
a.antonacopoulos@primaresearch.org

[3] University of Technology Sydney, Sydney, Australia
michael.blumenstein@uts.edu.au

Abstract. Script identification of text in natural scene images is challenging due to complex backgrounds, arbitrary orientations, different-sized characters, varying fonts, and multiple styles. Most existing methods are not effective in the presence of the above challenges. This paper introduces a new approach based on the Xception architecture and employing the log-polar transformed original image as an additional input, enabling the extraction of cues that are invariant to rotation, scaling, but are sensitive to script. The rationale behind the proposed work is that the combination of global features with text style features makes a significant difference in discriminating between different scripts. To combine the features extracted by Xception from the input image and log the polar transform of the input image, the proposed method introduces a style-enhanced fusion block. In addition, to further improve the performance of script identification, the proposed approach uses a new receptive channel selective focal attention module. Comparative evaluation results on three benchmark datasets, namely CVSI 2015, SIW-13, and MLe2e show that the proposed method outperforms the state-of-the-art in terms of classification rate.

Keywords: Deep learning · Xception architecture · Log polar transform · Attention model · Scene text · Script identification

1 Introduction

Developing a universal Optical Character Recognizer for multiple scripts is not feasible nor advisable. The key reason is that the shape and characteristics of characters are not shared among multiple scripts. While, for instance, Kannada-Telugu, and Devanagari-Bengali characters share similar characteristics pairwise. Therefore, although there are successful OCR systems for recognizing documents in English and other Latin scripts,

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 183–198, 2025.
https://doi.org/10.1007/978-3-031-78495-8_12

developing a single OCR for multiple scripts is very difficult without losing accuracy. Thus, to recognize multiple scripts in natural scenes or in a document collection, it is essential for a script identification step to take place before an appropriate OCR can be selected for each particular script/language. Script identification is not a new problem in the field of document image analysis. While there are several methods in the literature, most are not effective for arbitrary rotations and scaling [1, 2]. Therefore, addressing the challenge of arbitrary orientation and scaling remains an open challenge. It is evident from the sample results of the proposed and existing methods, illustrated in Fig. 1, that the most prominent state-of-the-art approach [1] fails to recognize the script of Kannada, Korean, and Chinese. On the other hand, the proposed method identifies all the scripts shown in Fig. 1 accurately.

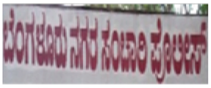
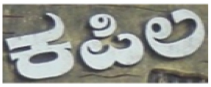
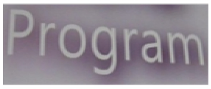
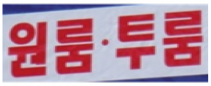
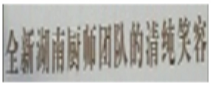
Image	Ground Truth	FAS-Res2Net	Our model
	Kannada	Chinese	Kannada
	Kannada	Oriya	Kannada
	Latin	Latin	Latin
	English	English	English
	Korean	Chinese	Korean
	Chinese	Korean	Chinese

Fig. 1. Sample images where FAS-Res2net [1] fails to predict correctly, whereas our model performs well.

The existing FAS-Res2net [1] method effectively captures deep semantic spatial global and local features for script identification by leveraging features of diverse depth and scale. However, it is still not sufficient to identify some of the scripts that share almost the same characteristics. This is due to the fact that the method is not robust to orientation and scaling. To address this challenge, the proposed approach integrates features from log-polar transformed images. This integration enables the model to capture scale and rotational variations as alterations in the x and y directions, which are effectively accommodated by convolutional layers. Thus, the proposed method offers a solution to the inherent limitations of Convolutional Neural Networks (CNNs) in handling scale and rotational variations, enhancing its robustness in script identification.

The key contributions of the work described in this paper are as follows. (i) Proposing the Xception architecture for global feature extraction is new compared to the state-of-the-art methods. (ii) Exploring the log-polar transform to make the proposed model invariant to rotation and scaling is new. (iii) The way the proposed work fuses the global features with the features extracted from log-polar transformed images using a style-enhanced fusion block and a receptive channel selective focal attention module is also a novel idea.

In the remainder of this paper, a review of script identification methods is presented in Sect. 2. The Xception architecture and the other steps of the proposed method are described in Sect. 3. Section 4 presents and discusses experimental results, while Sect. 5 offers concluding remarks.

2 Related Work

Significant advancements have been made in script identification within scene images, leading to the development of several methods. Bunia et al. [3] proposed a model specifically for Indian handwritten script identification, focusing on character-level feature extraction. However, this work is confined to Indian handwritten scripts and does not address scripts within scene images. Gomez et al. [2] developed a scene script identification model based on patch-based analysis, dividing input images into patches and calculating global loss using multiple patches rather than individual images. Cheng et al. [4] concentrated on combining CNNs and a Patch Aggregator (PA) for script identification, utilizing prediction scores from local patches.

Cheikhrouhou et al. [5] aimed to build an end-to-end multi-task deep neural network for script identification and spotting in document images, limited to handwritten document images, not scene images. Dutta et al. [6] explored the inception network for extracting global and local features for script identification within scene images. Khalil et al. [7] introduced an end-to-end model for text detection and script identification in scenes, with performance dependent on successful text detection. Bhunia et al. [8] focused on both scene and video script identification, leveraging CNNs and LSTMs for local and global feature extraction.

Guo et al. [9] proposed a combination of deep convolutional neural networks and spatial pyramid pooling for identifying scripts in ancient books. Li et al. [10] introduced a self-attention network for scene script identification, while Shivakumara et al. [11] used conventional features to address video script challenges. Udupa et al. [12] explored YOLOv5 for text localization and script identification and tested it on custom datasets. Shi et al. [13] proposed CNNs for deep and mid-level feature extraction, and Lu et al. [14] utilized CNNs and attention CNNs for discriminative patch mining.

Mahajan et al. [15] concentrated on Indian script identification using CNN enhancement, whilst Mei et al. [16] explored CNNs and RNNs for scene script identification, and Ma et al. [17] developed hierarchical feature fusion blocks and attention mechanisms. Shivakumara et al. [18] explored conventional gradient angular features for script identification, while Zhang et al. [1]] proposed Res2net for scene script identification. Yang et al. [19] developed end-to-end models for detection and identification, combining visual and textual features.

In summary, while most methods combine local and global features for script identification, they often lack robustness against arbitrary orientation and scaling, common in scene text. The proposed work utilizes of log-polar transform to enhance robustness to rotation and scaling, setting it apart from existing state-of-the-art script identification approaches.

3 Proposed Method

The proposed architecture leverages a customized adaptation of the Xception architecture, designed to enhance its performance in handling images. The model receives two inputs: the original image with dimensions 256 × 256 × 3 and its log-polar transformed counterpart. This log-polar transformation plays a pivotal role in addressing the inherent challenges of convolutional neural networks (CNNs) regarding rotation and scale variations. The model's initial layers extract features from both the input image in rectangular coordinates and the log-polar transformed image.

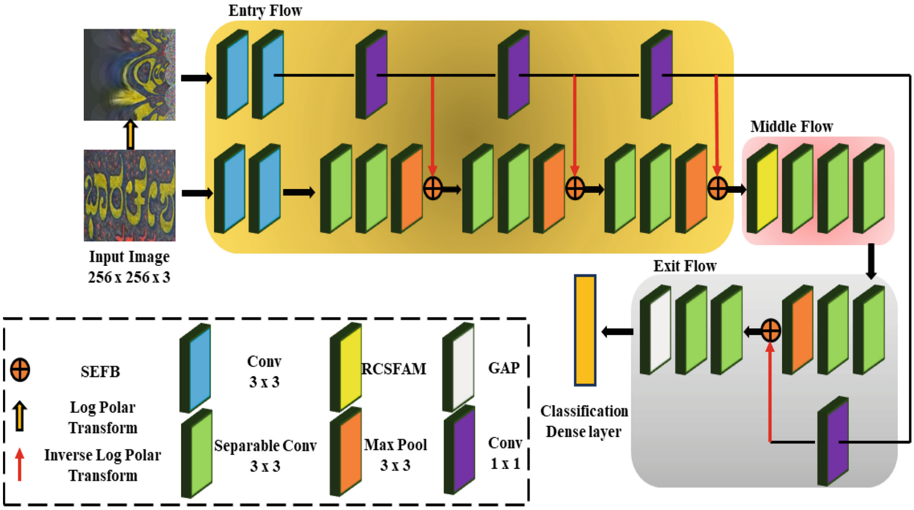


Fig. 2. The proposed architecture.

These features are subsequently fused using the Style Enhanced Fusion Block (SEFB). By combining information from both representations, the model gains robustness in dealing with diverse spatial transformations. To further improve the model's ability to discern important features, the Receptive Channel Selective Focal Attention Module (RCSFAM) is introduced. This module selectively emphasizes relevant channels while discarding others, enhancing the network's focus on crucial information. This attention mechanism proves crucial in addressing the challenges posed by variations in rotation and scale. The complete framework of the proposed method is presented in Fig. 2.

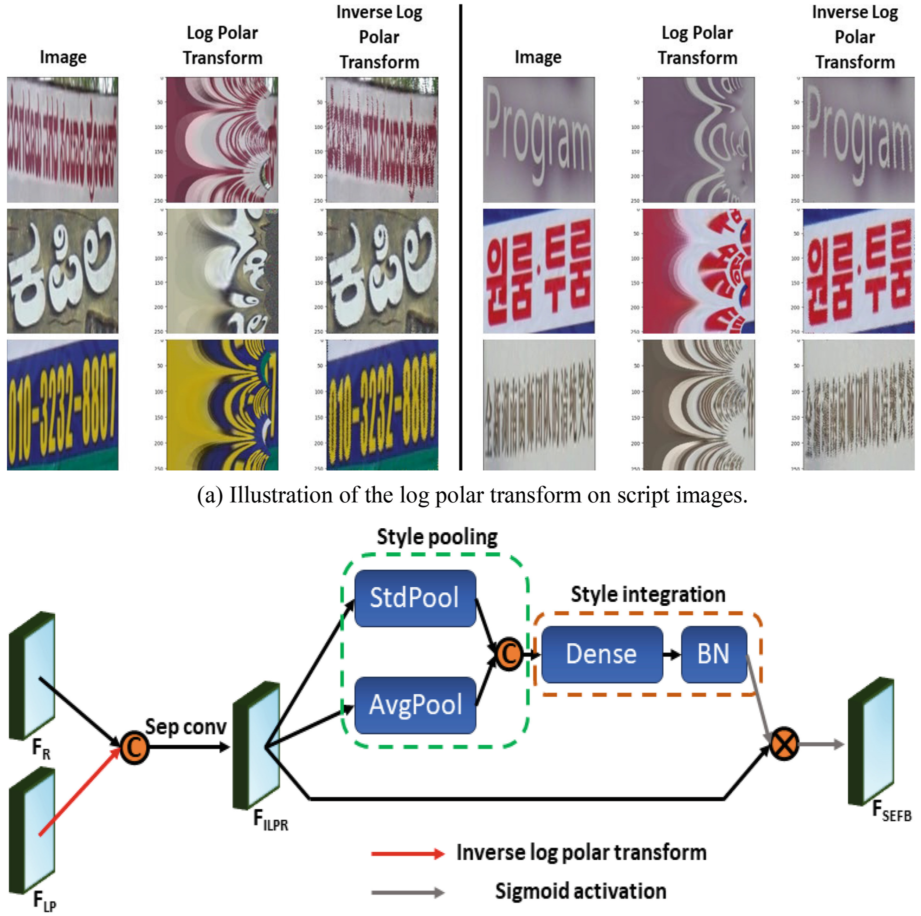
3.1 Log Polar Transform Feature and Style Enhanced Feature Block (SEFB)

The log-polar transform stands as a potent image processing technique that converts Cartesian coordinates (x, y) to polar coordinates (ρ, θ), creating a distinctive representation of visual information. The transformation is mathematically articulated as $\rho = \log\sqrt[2]{(x - x_c)^2 + (y - y_c)^2}$, encapsulating details of radial distances, while $\arctan2(y - y_c, x - x_c)$ captures angular intricacies ((x_c, y_c) are the coordinates of the image center). This duality imparts to the network the capacity to discern and categorize scripts not just based on their shapes but also their orientation. In the realm of CNNs, particularly sensitive to translation but less so to rotation and scale, the log-polar transform emerges as an invaluable asset. It adeptly navigates the hurdles presented by variations in script orientation and size by furnishing a more equivariant representation. As a result, the log-polar transform elevates the CNN's proficiency in recognizing and classifying scripts across diverse rotations and scales, establishing itself as a very effective preprocessing step in the script identification landscape. This transformation process enables the network to represent features in a robust and invariant way, contributing to higher accuracy and generalization in script identification.

The effectiveness of the Log-Polar transform can be observed in Fig. 3, where one can see the output of the inverse log-polar transform is almost the same as the input image for all the scripts listed in Fig. 3(a). This infers that the angle and radial information are capable of preserving the finer details of the content with additional rotation and scaling invariance features.

The Style Enhanced Fusion Block (SEFB) fuses the rectangular features and the features from the log-polar transformed image. F_R of dimension H × W × C are the features of the original image (in a rectangular coordinate system) whereas F_{LP} of dimension H W × C is the extracted feature from the log-polar transformed image. F_{LP} is converted to the Cartesian coordinate system (by applying the inverse log-polar transform).

This converted F_{LP} is then concatenated with F_R to generate a feature map of dimension H × W × 2C and convolved by a separable convolutional layer (C number of filters and kernel size of 1) to produce F_{ILPR} of dimension H × W × C. An overview of the SFEB module can be seen in Fig. 3(b) where one can see two key components: style integration and style pooling. In the style pooling part, channel-wise statistics are utilized, inspired by [20]. The proposed model calculates the statistical average value μ_{bc} and standard deviation μ_{bc} of each channel of F_{ILPR} which are represented as StdPool and AvgPool in Fig. 3(b). They are concatenated together to produce a style pooled feature of dimension H × W × 2. The style integration part comprises a channel-wise fully connected Dense layer, followed by a batch normalization (BN) layer, and a sigmoid activation function. By a style integration unit, the style pooled features can automatically learn style weights at the channel level. Finally, this style integrated feature of dimension H × W × 1 is then multiplied with F_{ILPR} to produce F_{SEFB} of dimension H × W × C.



(a) Illustration of the log polar transform on script images.

(b) Style Enhanced Fusion Block (SEFB).

Fig. 3. Log polar transform and the SEFB module.

3.2 Receptive Channel Selective Focal Attention Module (RCSFAM)

The Receptive Channel Selective Focal Attention Module (RCSFAM) is a modified version of the Focal Attention Module [21], which utilizes the information of the various receptive fields as shown in Fig. 4. It focuses on the relevant and important feature channels to enrich the middle flow of the Xception architecture with attention-aided useful features. Focal Modulation offers several advantages over traditional self-attention mechanisms, particularly in the context of script identification tasks using CNNs. The advantages are:

Computational Efficiency: Focal Modulation $\mathrm{y_i} = \mathrm{T_2}(\mathrm{M_2}(\mathrm{i};\ \mathrm{X});\ \mathrm{x_i})$ involves an early aggregation procedure where the context features are first aggregated using $\mathrm{M_2}$ at each location *i*, and then the query interacts with the aggregated feature based on $\mathrm{T_2}$ to form $\mathrm{y_i}$.

In contrast, self-attention requires summing over non-shareable attention scores for different queries, resulting in a computationally expensive process $y_i = T_1(M_1(x_i; X); X)$. The interaction T_2 in Focal Modulation is a lightweight operator between a token and its context, while T_1 in self-attention involves computing token-to-token attention scores, which have quadratic complexity.

Spatial and Channel-Specific Modulation: Focal Modulation allows for spatial and channel-specific modulation, enhancing the model's ability to discern relevant features $y_i = q(x_i) \odot m(i; X)$. This spatial and channel specificity is achieved through the element-wise multiplication operation, enabling selective modulation of token features.

Translation Invariance: Focal Modulation is inherently translation invariant due to its centering at the query token *i* and the absence of positional embeddings in the modulation process. This translation invariance ensures consistent performance across different spatial shifts, a crucial aspect in script identification tasks.

Explicit Input Dependency: Focal Modulation explicitly depends on the input context features X by aggregating local features around the target location *i*. This input dependency enables Focal Modulation to capture fine-grained spatial and semantic information, contributing to improved script identification accuracy.

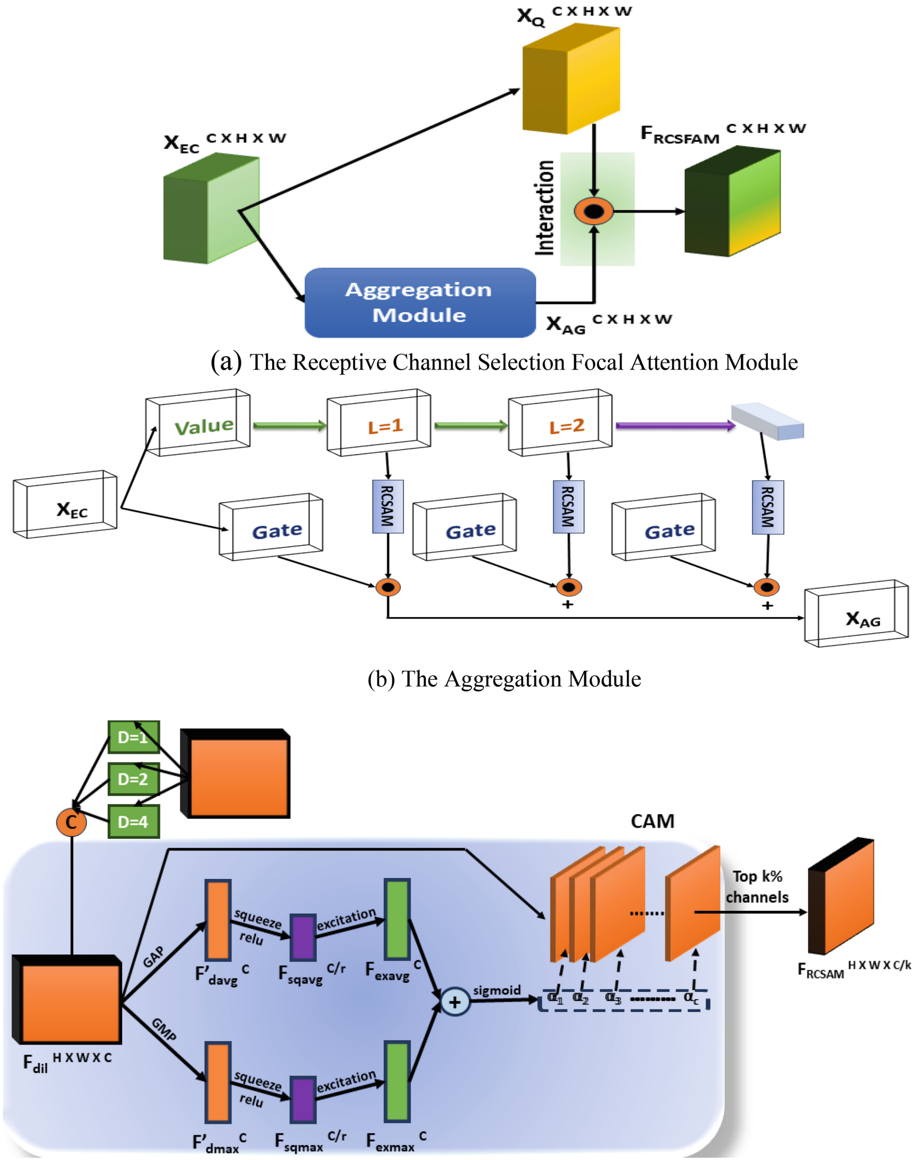
Thus, the Focal Modulation's efficiency, reduced complexity, spatial and channel specificity, translation invariance, and explicit input dependency make it a superior choice over self-attention for script identification tasks using CNNs. These advantages facilitate more effective feature refinement and context integration, leading to enhanced model performance and robustness in identifying and classifying scripts.

The Aggregation module of the Focal Attention Module shown in Fig. 4(a)-(b) is modified to further select the relevant channels to be multiplied using the gates of the gated aggregation. The Receptive Channel Selective Attention Module (RCSAM) is applied to the features of levels 1 and 2 ($l = 1$ and $l = 2$) as shown in Fig. 4(b). The features from $l = 1$ and 2 are convolved using layers of dilations = 1,2, and 4. These dilated features are concatenated to generate F_{dil}. These dilated features capture information from a wider range of receptive fields. The most important component of RCSAM, i.e. the Channel Attention Module (CAM), aims to assign weights (α) to the channels of concatenated feature maps to amplify those channels that contribute most significantly to enhancing model performance. The resulting output channel weights, denoted as α, have dimensions $C \times 1 \times 1$, where C represents the number of channels. α is a composite of two distinct features, F_{exavg} and F_{exmax}, which are defined in Eq. (1) and Eq. (2).

$$F_{exavg} = D_2(\mathrm{ReLU}(D_1(\mathrm{GAP}(F_{dil})))) \tag{1}$$

$$F_{exmax} = D_2(\mathrm{ReLU}(D_1(\mathrm{GMP}(F_{dil})))) \tag{2}$$

Here, D_1 and D_2 denote Dense layers with units C/r and C, respectively. The parameter r signifies the extent to which the features are compressed to C/r from C before being expanded back to their original dimension of C. This compression and expansion process



(a) The Receptive Channel Selection Focal Attention Module

(b) The Aggregation Module

(c) The Receptive Channel Selective Attention Module

Fig. 4. Illustration of the RCSFAM module and its components.

is commonly referred to as squeeze and excitation attention. The weights α can be computed using $\mathrm{F}_{\mathrm{exavg}}$ and $\mathrm{F}_{\mathrm{exmax}}$ according to the following Eq. (3).

$$\alpha = 6\left(\mathrm{F}_{\mathrm{exmax}} + F_{exavg}\right) \tag{3}$$

Here, 6 represents the sigmoid activation function, and + denotes element-wise addition. The combination of $\mathrm{F}_{\mathrm{exmax}}$ and F_{exavg} facilitates the identification and amplification of channels that are most influential in enhancing model performance. Thus, the channels of F_{dil} are sorted according to the values of the corresponding $\alpha(\alpha1, \alpha2, \ldots)$. The top k% channels from F_{dil} are selected to generate F_{RCSAM} of dimensions H × W × kC. The effectiveness of RCSFAM is shown in Fig. 5(a)–(c) where it is noted that the heat maps of different scripts of three datasets highlight the vital region in the input image for extracting distinct features for identification of scripts.

4 Experimental Results

Benchmark datasets have been used for experimentation, namely CVSI2015, SIW-13, and MLe2e. The details of these datasets, including the number of samples used for training, validation, and testing are summarized in Table 1.

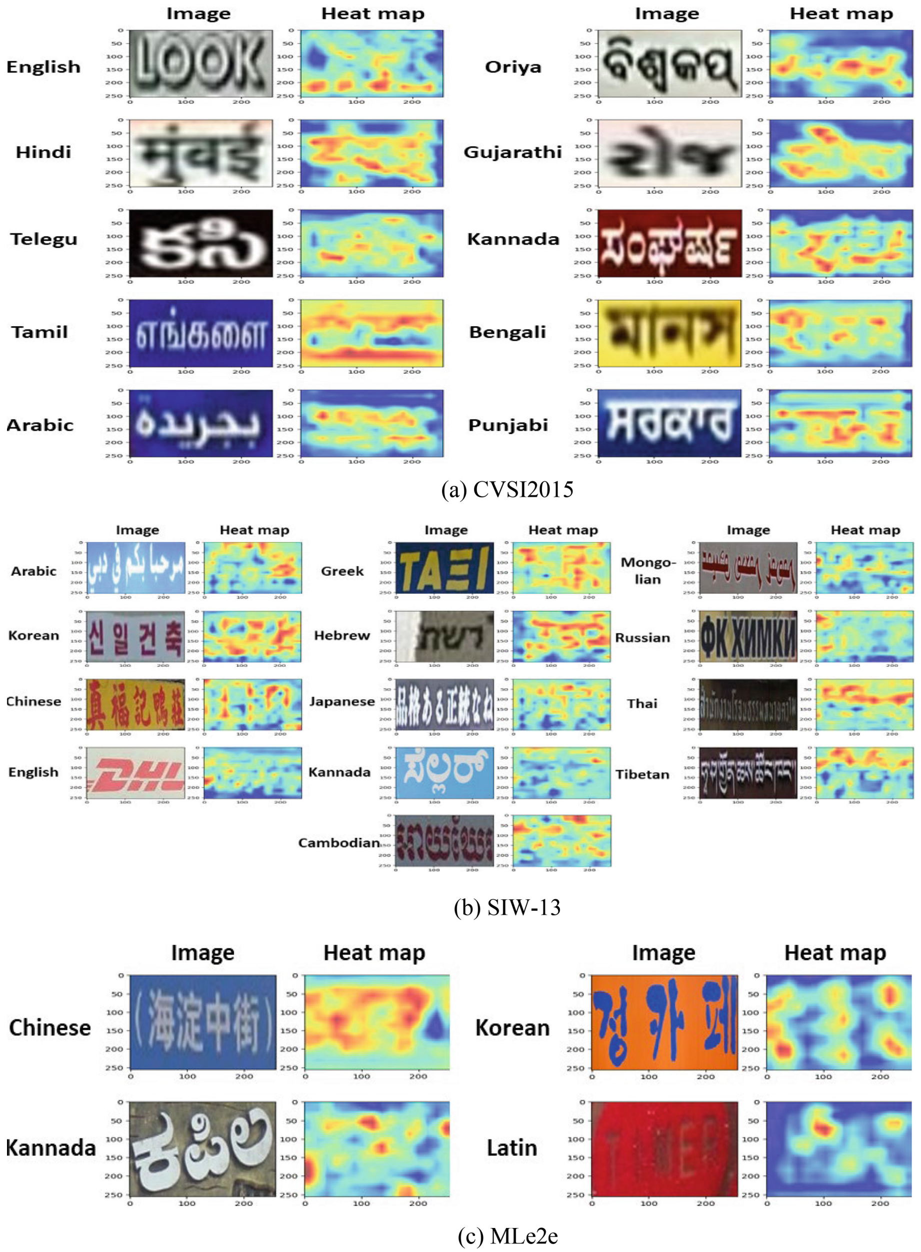
CVSI2015 Dataset [22]: This dataset comprises word images from various scripts extracted from videos, designed for script identification. It includes words from ten distinct scripts: English, Hindi, Bengali, Oriya, Gujarati, Punjabi, Kannada, Tamil, Telugu, and Arabic. **SIW-13 dataset** [23]: The SIW-13 dataset offers word-level images representing 13 different scripts. The scripts included in this dataset are Arabic, Cambodian, Chinese, English, Greek, Hebrew, Japanese, Kannada, Korean, Mongolian, Russian, Thai, and Tibetan. **MLe2e dataset** [2]: MLe2e is a popular and standard script identification dataset comprising a total of 711 scene images covering four different scripts (Latin, Chinese, Kannada, and Hangul).

To measure the performance of the proposed method against the state-of-the-art, the standard measures of Accuracy, Precision, Recall, and F1-score are used.

Implementation Details: The model is trained for 50 epochs, with augmentation of contrast enhancement by a factor of 1.5. The Learning rate was set to 0.01 with an Adam optimizer [24]. The TensorFlow-wavelets library was used for wavelet transformation. A system with an Intel Core i7 processor, with 8GB RAM was used for training and a system with an NVIDIA P100 GPU. Python version 3.7.4 is used for implementation. Cross entropy loss [24] is used for the training of all the branches.

4.1 Ablation Study

To individually evaluate the contributions of the key steps of the proposed method, namely, the Xception architecture, the log-polar transform and the optimal value of *k*, a series of ablation experiments were conducted on the CVSI2015 dataset, as presented in Table 2. The experiments (i) and (ii) show that Xception alone does not achieve the best results when it is compared with the Xception + Log-Polar Transform. This infers that leveraging features from log-polar transformed images significantly improves the performance of the proposed method. Introducing the Style Enhanced Fusion Block (SEFB) to effectively fuse rectangular and log-polar features further elevates the model's performance as reported in experiment (iii). Subsequently, integrating the Focal Attention Module (FAM) leads to additional performance gains as reported in experiment (iv). Experiment (v) shows that the proposed model achieves the best accuracy compared



(a) CVSI2015

(b) SIW-13

(c) MLe2e

Fig. 5. Heatmaps of RCSFAM.

to all the experiments on individual steps. Therefore, one can conclude that all the key steps are effective for achieving the best performance of the proposed method.

Table 1. Details of training, testing, and validation images from all three datasets

Dataset	Class	Training	Testing	Validation
CVSI2015	10	6412	3234	1069
SIW-13	13	9103	3299	3889
MLe2e	4	826	642	351

To delve deeper into the RCSFAM architecture, experiments were conducted to determine the optimal value of k. As depicted in Table 3, the model achieves its peak performance with $k = 40\%$. Therefore, the proposed method sets the value of k as 40%. Moreover, the impact of log-polar features across different classes is evident from Table 4. To validate the contribution of the log-polar transform, we conducted additional experiments for all the classes of the CVSI2015 dataset as reported in Table 4. It is noted from Table 4, for all the classes, the results of the proposed method with log-polar transform are better than the results of the proposed method without log-polar transform. It should be noted that for all the ablation experiments, the CVSI2015 dataset has been used as it is the most widely used and the images are collected from video (rather than being still images), and those video images indeed have variations in terms of low contrast, quality and background compared to other datasets.

Table 2. Analyzing the effect of different components of the proposed model on its performance on the CVSI2015 dataset

#	Method	Accuracy	Precision	Recall	F-score
(i)	Xception	93.97	94.25	93.58	93.91
(ii)	Xception + Log Polar images	96.30	97.12	94.99	95.88
(iii)	Xception + Log Polar images + SEFB	96.83	97.16	95.12	96.17
(iv)	Xception + FAM	98.18	98.51	98.49	98.43
(v)	Xception + RCSFAM (k = 40%)	98.48	98.55	98.78	98.49

4.2 Comparison with the State-of-the-Art

The confusion matrices of the proposed method for the classification on all the three datasets are shown in Fig. 6, where it is noted that for all the classes, our method achieves promising performance. This indicates that the proposed method is stable and reliable across classes and datasets. To validate the usefulness of the proposed method, quantitative results of the proposed and state-of-the-art methods on all three datasets are reported in Table 5, where one can see that the proposed method is superior compared to the state-of-the-art. The key reason for the poor results of the existing methods is that, as discussed in the proposed methodology section, those methods are not effective in

Table 3. Analyzing the effect of *k* in RCSFAM on the CVSI2015 dataset

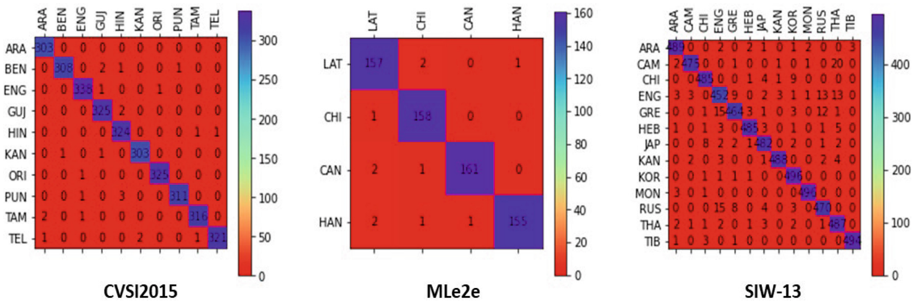
Method	Accuracy	Precision	Recall	F1-score
k = 30%	98.56	98.51	97.19	96.92
k = 40%	98.99	98.75	98.37	98.83
k = 50%	98.48	98.55	98.78	98.49
k = 60%	98.16	98.35	98.23	98.15
k = 70%	98.05	98.17	98.01	97.99

Table 4. Studying the effect of the log-polar transform on the CVSI2015 dataset.

Scripts	Without log-polar transformed image	With log-polar transformed image
Arabic	92.09	96.28
Bengali	83.10	86.33
English	88.61	89.20
Gujarati	89.33	94.46
Hindi	91.29	94.48
Kannada	79.34	81.76
Oriya	91.67	93.49
Punjabi	91.51	94.30
Tamil	83.12	85.39
Telugu	81.22	81.74

the presence of arbitrary orientations and scaling. But in the case of scene text, arbitrary orientation and scaling are common and therefore, it is essential to address these two challenges to improve the performance of script identification. As noted from the ablation study, introducing the log-polar transform makes a significant impact on improving the performance of the proposed method for script identification on all three datasets.

To test the robustness of the proposed method and to show the proposed method is invariant to rotation and scaling, the accuracy of the method was calculated for randomly rotated and scaled images of the CVSI2015 dataset, as reported in Table 6. The results show that the performance of the proposed method is better than the state-of-the-art on both rotated and scaled images. It is also noted that the results obtained by the proposed method for different rotations and scaling are almost similar to the results for normal images. Thus, one can assert that the proposed method is robust to rotations and scaling.



CVSI2015

	ARA	BEN	ENG	GUJ	HIN	KAN	ORI	PUN	TAM	TEL
ARA	303	0	0	0	0	0	0	0	0	0
BEN	0	308	0	2	1	0	0	1	0	0
ENG	0	0	338	1	0	0	1	0	0	0
GUJ	0	0	0	325	2	0	0	0	0	0
HIN	0	0	0	0	324	0	0	0	1	1
KAN	0	1	0	1	0	303	0	0	0	0
ORI	0	0	1	0	0	0	325	0	0	0
PUN	0	0	1	0	3	0	0	311	0	0
TAM	2	0	1	0	0	0	0	0	316	0
TEL	1	0	0	0	0	2	0	0	1	321

MLe2e

	LAT	CHI	CAN	HAN
LAT	157	2	0	1
CHI	1	158	0	0
CAN	2	1	161	0
HAN	2	1	1	155

SIW-13

	ARA	CAM	CHI	ENG	GRE	HEB	JAP	KAN	KOR	MON	RUS	THA	TIB
ARA	489	0	0	2	0	2	1	0	1	2	0	0	3
CAM	2	475	0	0	1	0	0	1	0	1	0	20	0
CHI	0	0	485	0	0	1	4	1	9	0	0	0	0
ENG	3	3	0	452	9	0	2	3	1	1	13	13	0
GRE	0	0	1	15	464	3	1	0	3	0	12	1	0
HEB	1	0	1	3	0	485	3	0	1	0	1	5	0
JAP	0	0	8	2	2	1	482	0	2	0	1	2	0
KAN	0	2	0	3	0	0	1	488	0	0	2	4	0
KOR	0	0	1	1	1	1	0	0	496	0	0	0	0
MON	3	0	1	0	0	0	0	0	0	496	0	0	0
RUS	0	0	0	15	8	0	4	0	3	0	470	0	0
THA	2	1	1	2	0	1	3	0	2	0	1	487	0
TIB	1	0	3	0	1	0	0	0	0	0	0	0	494

Fig. 6. Confusion matrix of the proposed model on CVSI2015, Mle2e, and SIW-13.

Table 5. Comparison of the proposed model with the SOTA models on the CVSI2015, ML22e, and SIW-13 datasets

Method	Accuracy (%)		
	CVSI2015	SIW-13	MLe2e
Mei [16]	94.20	92.75	–
Shi [13]	94.30	89.40	–
Gomez [2]	97.20	94.80	94.40
Bhunia [8]	97.75	–	96.70
Cheng [4]	98.60	96.50	–
Ma [17]	98.78	97.30	97.20
Lu [14]	97.90	96.11	89.42
Dutta [6]	98.97	95.70	95.01
Mahajan [15]	97.40	–	–
Guo [9]	93.50	–	98.74
FAS-Res2net [1]	96.00	94.70	–
Proposed	**98.99**	**97.83**	**98.75**

Table 6. Performance of the proposed method and existing method [1] on randomly rotated and scaled images from the CVSI2015 dataset

Method	Rotation (–40 to +40)	Scaling (–1.5 to +1.5)
Proposed	94.26	97.03
FAS-Res2net [1]	89.98	94.67

4.3 Limitations

While the proposed model demonstrates robustness against scale and rotational variations, it encounters challenges in accurately classifying instances where the scripts appear exceedingly hazy or blurred against the background, resulting in unclear script visibility. The example images illustrated in Fig. 7 and the accompanying heatmaps indicate that the model has difficulty in scenarios characterized by hazy or unclear script images. This shows that there is scope for improvement, albeit this is not a straightforward problem by normal standards. To overcome this problem, the authors plan to explore the integration of language models in the proposed approach.

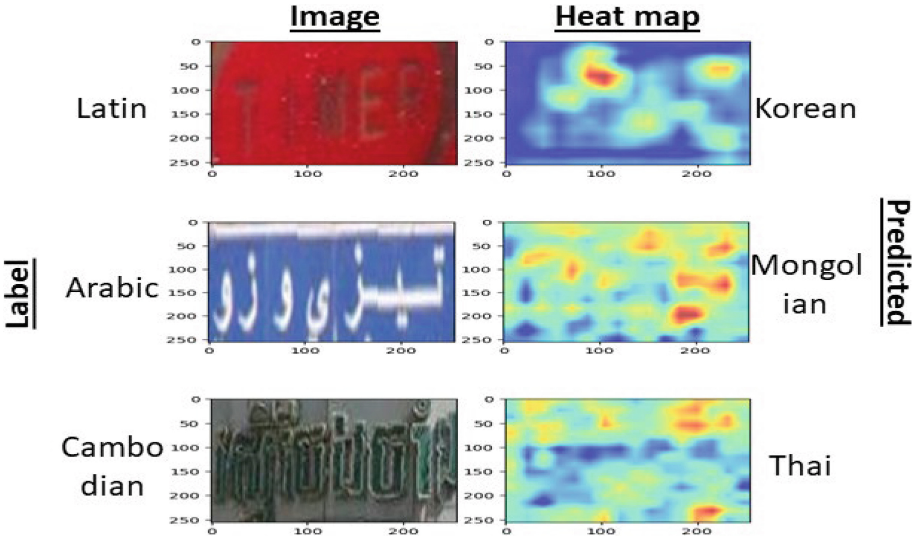


Fig. 7. The error cases where the proposed model misclassifies a script.

5 Conclusions and Future Work

A novel method for script identification in natural scene images has been proposed in this paper. Unlike state-of-the-art approaches, which do not deal with the challenges of arbitrary orientation and scaling, the proposed work introduces the use of the log-polar transform to make the method robust to rotation and scaling to improve script identification performance. In addition, the proposed work makes use of the Xception architecture for feature extraction from both the input image and the log-polar transform of the input image. To strengthen the features, a style-enhanced fusion block and a receptive channel selective focal attention module are proposed. Comprehensive experiments were conducted to evaluate the proposed approach, including an ablation study to validate the key steps and the proposed method. The results of comparative evaluation reveal that the proposed method outperforms the state-of-the-art on all three benchmark datasets. However, when the images suffer from severe blur and haze, the performance of the method degrades. This could be addressed by integrating language models in the proposed model, a future work direction planned by the authors.

References

1. Zhang, Z., Mamat, H., Xu, X., Aysa, A., Ubul, K.: FAS-Res2net: an improved Res2net-based script identification method for natural scenes. Appl. Sci. **13**(7), 4434 (2023)
2. Gomez, L., Nicolaou, A., Karatzas, D.: Improving patch-based scene text script identification with ensembles of conjoined networks. Pattern Recogn. **67**, 85–96 (2017)
3. Bhunia, A.K., Mukherjee, S., Sain, A., Bhunia, A.K., Roy, P.P., Pal, U.: Indic handwritten script identification using offline-online multi-modal deep network. Inf. Fusion 1–14 (2020)
4. Cheng, C., Huang, A., Bai, X., Feng, B., Liu, W.: Patch aggregator for scene text script identification, In: Proceedings ICDAR, pp. 1077–1083 (2019). https://doi.ieeecomputersociety.org/10.1109/ICDAR.2019.00175
5. Cheikhrouhou, A., Kessenini, Y., Kanoun, S.: Multi-task learning for simultaneous script identification and keyword spotting in document images. Pattern Recogn. **113** (2021)
6. Dutta, K., Dastidar, S.G., Das, N., Kundu, M., Nasipuri, M.: Script identification in natural scene text images by learning local and global features on inception net, In Proceedings of the CVIP, pp. 458–467 (2022)
7. Khalil, A., Jarrah, M., Al-Ayyub, M., Jaraweh, Y.: Text detection and script identification in natural scene images using deep learning. Comput. Electr. Eng. **91** (2021)
8. Bhunia, A.K., Konwer, A., Bhunia, A.K., Bhowmick, A., Roy, P.P., Pal, U.: Script identification in natural scene image and video frames using an attention based convolutional-LSTM network. Pattern Recognit. **85**, 172–184 (2019)
9. Guo, H., Yang, D., Liu, Y., Zhao, J.: Script identification of ancient books by Chinese ethnic minorities using multi-branch DCNN and SPP. Pattern Anal. Appl. **26**(2), 809–821 (2023)
10. Li, X., Zhan, H., Shivakumara, P., Pal, U., Lu, Y.: SANet-SI: a new self-attention-network for script identification in scene images. Pattern Recogn. Lett. **171**, 45–52 (2023)
11. Shivakumara, P., Sharma, N., Pal, U., Blumenstein, M., Tan, C.L.: Gradient-angular-features for word-wise video script identification. In: Proceedings ICPR, pp. 3098–3103 (2014)
12. Udupa, C., Upadhyaya, A., Patil. B.S., Seeri, S.V., Patil, P., Hiremath, P.: Text localization and script identification in natural scene images and videos. In: Proceedings of CSI (2022)
13. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2016)
14. Lu, L., Wu, D., Tang, Z., Yi, Y., Huang, F.: Mining discriminative patches for script identification in natural scene images. J. Intell. Fuzzy Syst. **40**(1), 551–563 (2021)
15. Mahajan, S., Rani, R.: Word level script identification using convolutional neural network enhancement for scenic images. Trans. Asian Low-Resour. Lang. Inf. Process. **21**(4), 1–29 (2022)
16. Mei, J., Dai, L., Shi, B., Bai, X.: Scene text script identification with convolutional recurrent neural networks. In: Proceedings of ICPR, pp. 4053–4058 (2016)
17. Ma, M., Wang, Q.F., Huang, S., Huang, S., Goulermas, Y., Huang, K.: Residual attention-based multi-scale script identification in scene text images. Neurocomputing **421**, 222–233 (2021)
18. Shivakumara, P., Yuan, Z., Zhao, D., Lu, T., Tan, C.L.: New gradient-spatial-structural features for video script identification. Comput. Vision Image Underst. 35–53 (2015)

19. Yang, K., Yi. J., Chen, A., Liu, J., Chen, W., Jin, Z.: ConvPatchTrans: A script identification network with global and local semantics deeply integrated. Eng. Appl. Artif. Intell. **113** (2022)
20. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the ICCV, pp. 1501–1510 (2017)
21. Yang, J., Li, C., Dai, X., Gao, J.: Focal modulation networks. Adv. Neural. Inf. Process. Syst. **35**, 4203–4217 (2022)
22. Sharma, N., Mandal, R., Sharma, R., Pal, U., Blumenstein, M.: ICDAR2015 competition on video script identification (CVSI 2015). In: Proceedings of ICDAR, pp. 1196–1200 (2015)
23. Shi, B., Bai, X., Yao, C.: Script identification in the wild via discriminative convolutional neural network. Pattern Recogn. **52**, 448–458 (2016)
24. Hui, L., Belkin, M.: Evaluation of neural architectures trained with square loss vs cross-entropy in classification tasks. arXiv preprint arXiv:2006.07322 (2020)

DITS: A New Domain Independent Text Spotter

Kunal Purkayastha[1,3], Shashwat Sarkar[1], Palaiahnakote Shivakumara[2](✉), Umapada Pal[1], Palash Ghosal[3], and Xiao-Jun Wu[4]

[1] Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, Kolkata, Baranagar, India
umapada@isical.ac.in
[2] School of Science, Engineering and Environment, University of Salford, Manchester, UK
s.palaiahnakote@salford.ac.uk
[3] Department of Information Technology, Sikkim Manipal Institute of Technology, Sikkim Manipal University, Gangtok, Sikkim, India
[4] Jiangnan University, Wuxi, China
wu_xiaojun@jiangnan.edu.cn

Abstract. Text spotting in diverse domains, such as drone-captured images, underwater scenes, and natural scene images, presents unique challenges due to variations in image quality, contrast, text appearance, background complexity, and external factors like water surface reflections and weather conditions. While most existing approaches focus on text spotting in natural scene images, we propose a Domain-Independent Text Spotter (DITS) that effectively handles multiple domains. We innovatively combine the Real-ESRGAN, developed for regular image enhancement, with the DeepSolo, developed for scene text spotting, in an end-to-end fashion for text detection and spotting on images of different domains. The key idea behind our approach is that improving image quality and text-spotting accuracy are complementary goals. Real-ESRGAN enhances image quality, making the text more discernible, while DeepSolo, a state-of-the-art text spotting model, accurately localizes and recognizes text in the enhanced images. We validate the superiority of our proposed model by evaluating it on datasets from drone, underwater, and scene domains (ICDAR 2015, CTW1500, and Total-Text). Furthermore, we demonstrate the domain independence of our model through cross-domain validation, where we train on one domain and test on others. Our dataset and code will be publicly available on GitHub.

Keywords: Scene text detection · Scene text recognition · DeepSolo · Super resolution · Domain independent · Text spotting

1 Introduction

Addressing the problem of multiple domains is essential when we look at real-time and real-world applications. For example, we can see many activities, such as dining, games, capturing marriage moments, creating awareness, scuba diving, etc. [1]. Similarly, drones are popular for surveillance applications such as marathons, sports, exhibitions, and other

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 199–216, 2025.
https://doi.org/10.1007/978-3-031-78495-8_13

social events [2]. Scene images are helpful for automatically driving vehicles, locating exact places in the city, answering visual questions, etc. [3]. Therefore, it is evident that input images can be of any domain rather than from one domain. However, most existing methods developed so far focus on one domain or particular dataset [1–3]. For instance, some methods focus mainly on the scene, drone, or underwater images for text detection. Therefore, the methods developed for a particular dataset and domain may not be effective for multiple domain datasets. The critical problem for the inability to work on images of multiple domains is that each domain poses different challenges. For example, underwater images suffer from poor quality, visibility, low contrast, and illumination effect, while drone images suffer from tiny text because of the height distance of the drone, loss of information due to occlusion, and scene images suffer from arbitrarily oriented/shaped text as shown in Fig. 1.

It is evident from the results presented in Fig. 1 that the existing method, which is a state-of-the-art method for text spotting, does not work well for underwater and drone images, but it works well for scene images [3]. This shows that the developed scene text method works for particular types of images. On the other hand, the proposed method detects and recognizes well for the images of three domains.

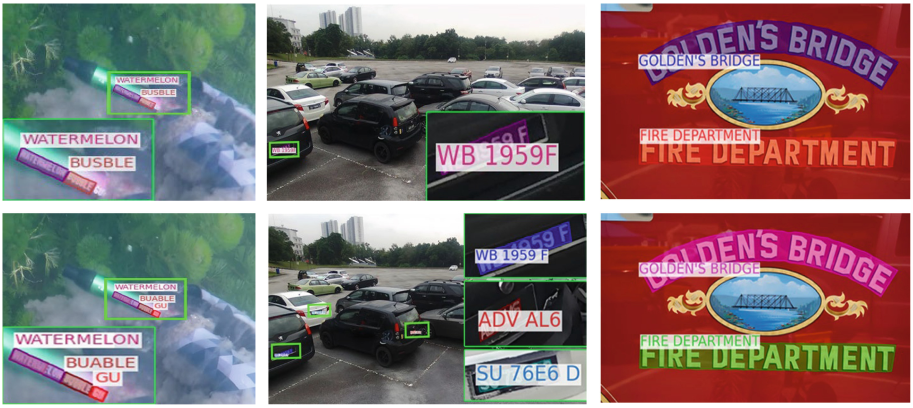


Fig. 1. Illustrating challenges of text spotting in underwater and drone images where the first row's images are the Performance of a latest state-of-the-art model [3] on images of different domains (Left column: underwater; Middle column: drone; Right column: scene). And the second row's images are the performance of the proposed method on multiple domains.

Therefore, this work proposes a novel model for text spotting across various domains, such as underwater, drone, and scene images. Drawing inspiration from Real-ESRGAN [4] and its effectiveness on images of varying quality, we propose to enhance the poor-quality images while maintaining the fine details of the text information. Similarly, the success of transformers in text spotting [3] has led us to explore DeepSolo, utilizing ResNet-50 as the backbone of our work.

It is true that Real-ESRGAN and DeepSolo are not new for image enhancement and text spotting. However, the scope of the Real-ESRGAN model is limited to general images, while the DeepSolo is limited to scene images. In addition, it is unclear whether these models can handle different domains, where one can expect different nature and complexity of the problems. It is evident from the results presented in Fig. 1 that the DeepSolo does not perform well for underwater and drone images compared to scene images. Therefore, the question is how to make those models work well for images of different domains, including poor quality images caused by underwater and drones and good quality scene images. Thus, the contribution lies in adapting Real-ESRGAN and DeepSolo as an end-to-end model to make it domain independent, which is challenging. Thus, the key contributions to address this challenge are as follows.

(i) Exploring Real-ESRGAN to improve the visual quality of poor-quality images before text spotting without affecting high-quality images. (ii) Adopting DeepSolo for text spotting in multiple domains is compared to other state-of-the-art methods. (iii)The way this proposed work integrates the enhancement and text spotting model as an end-to-end model is novel.

The rest of the paper is organized as follows. The existing methods of text spotting in natural scene images are reviewed in Sect. 2. In Sect. 3, the architectures of the proposed model are presented. Experimental analyses are discussed in Sect. 4. Section 5 summarizes the outcome of the proposed method and discusses future work.

2 Related Work

Here, we review the methods related to image enhancement, text detection, recognition, and text spotting in natural scene images.

There are models for enhancing image quality such that the performance of text detection, recognition, and spotting improves, especially for low-quality and distorted images. For example, Jianqi Ma et al. [5] proposed an architecture that enhances text super-resolution by incorporating a recognition module and text prior interpreter for better coherence between text and image features. Minglong Xue et al. [6] introduced the Attention Enhanced Residual-in-Residual Dense Network (AERRDN), which combines channel and spatial attention modules and uses gradient loss for improved edge restoration in text images. The review of image enhancement methods reveals that none of the techniques address underwater and drone images.

Similarly, several elegant models were proposed for text detection and recognition in natural scene images. For instance, Palaiahnakote Shivakumara et al. [7] developed a model for text detection and style transfer in social media images, using EffiUNet++ and TESP-Net to tackle challenges like low contrast. SPTSv2 [8] features a model for simultaneous text location and recognition with specialized decoders. Ayan Banerjee et al. [9] introduced a multi-view image text recognition model, integrating NLP and genetic algorithms. Jianqi Ma et al. [10] developed a method integrating text recognition priors into the super-resolution process, enhancing text clarity. These methods focus on natural scene images and do not cover complex underwater and drone images.

In the same way, many models are introduced for text spotting in natural scene images. The method called ABINet++ [11] uses a novel approach for scene text spotting, introducing Bezier curve detection for shape flexibility. Xixuan Hao et al. [12] introduced ARText, a unique Chinese text dataset emphasizing font and shape diversity, and presented Deformation Robust TextSpotter. ADATS [13] proposed a Transformer based on Adaptive RoI-Align for aspect-ratio-preserving text feature extraction, an attention-based segmentation head for accurate arbitrary-shaped text location. Taeho Kil et al. [14] introduced a unified approach for spotting arbitrary-shaped texts by integrating multiple detection formats into a single framework.

DSText V2 [15] enhances video text spotting research with dataset insights. DeepSolo [3] introduces a simplified DETR-like model with a single Transformer decoder for text spotting. IAST [16] proposes a framework for spotting various text types by integrating modules for reading order and dynamic sampling. FlowText [17] offers a novel approach to video text synthesis, addressing motion blur and occlusion. SwinTextSpotter [18] showcases the effectiveness of Transformer models in scene text spotting. ABCNetv2 [19] presents a breakthrough with its Bezier curve-based text representation and alignment for improved text spotting. GLASS [20] sets new benchmarks with its attention module and orientation loss. TESTR [21], inspired by DETR, employs a Transformer framework for text spotting. SPTS [22] uses single-point annotations for training text spotters, while DA-TextSpotter [23] excels in various environments with enhanced features for noisy images.

Recently, Banerjee et al. [1] pioneered text detection in underwater images using DCT, DWT, and FFT combined with a modified CRAFT model, enhancing text clarity and setting a new standard in underwater and natural scenes. Mokayed et al. [2, 24] developed a method for detecting license plate numbers in drone images using a discrete cosine transform and a phase congruency model with low contrast and quality enhancements. Pal et al. [25] utilized the Swin transformer for similar purposes in drone images, incorporating maximally stable external region analysis to reduce false positives. The above analysis shows that most models focus on a particular domain or dataset rather than multiple domains. Therefore, text spotting in multiple domains is an open challenge.

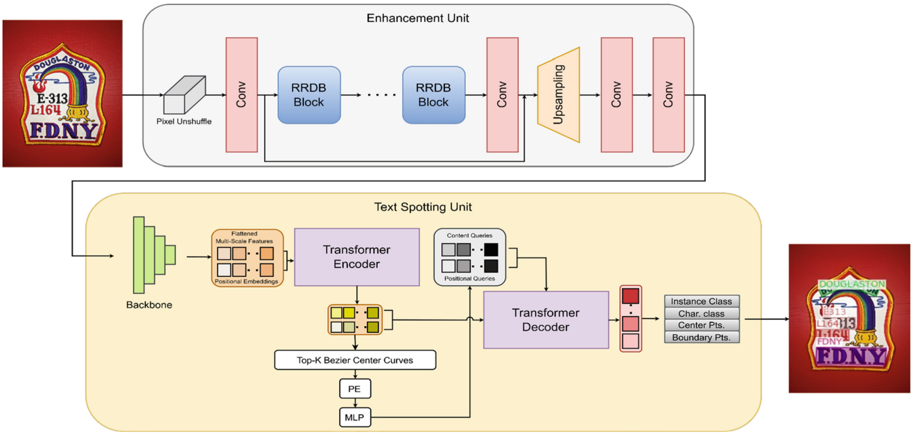


Fig. 2. Overall architecture of our proposed model. The architecture consists of an enhancement unit consisting of pixel unshuffled, convolution, RRDBNet and Upsampling operations further concatenated with a text spotting unit consisting of a feature extraction backbone followed by a Transformer Encoder and Decoderations.

3 Proposed Methodology

Our work strives to develop a domain-independent end-to-end text-spotting model for natural scenes, underwater, and drone images. It is noted from the previous section that each domain has its complexity and exhibits challenges. Inspired by the Real-ESRGAN (Enhanced Super-Resolution Generative Adversarial Network) [4], which is designed to improve the texture and edge details in low-contrast images, we use a similar approach to enhance the fine details specifically in underwater and drone images without affecting the fine details in high-quality images. In the same way, the accomplishments of transformers for text spotting in natural scene images motivated us to explore DeepSolo for text spotting in multiple domains in this work. Additionally, the proposed work integrates enhancement, detection, and recognition processes into a single end-to-end model. The full architecture of this model is illustrated in Fig. 2, showcasing the seamless integration of the enhancement model with text spotting model.

3.1 Enhancement Unit

The pixel unshuffle layer in the enhancement unit plays an essential role in the network's performance by reducing computational complexity, increasing the channel size, and improving the handling of real-world blind super-resolution tasks.

Pixel-unshuffle is an operation that does the inverse of pixel shuffle. In pixel shuffling, spatial resolution is increased by reorganizing the elements of the channel dimension into the spatial dimensions. Conversely, pixel-unshuffle reduces spatial dimensions by redistributing pixels into the channel dimension. This operation effectively decreases the height and width of the image while increasing the number of channels, thereby preparing the image for further processing. This process allows the network to upsample the feature map efficiently since most of the calculations are carried out at a lower resolution, which helps to decrease the consumption of GPU memory and computational resources. By employing the pixel unshuffle layer, Real-ESRGAN can gradually increase the spatial resolution of the feature maps as they pass through the network, ultimately resulting in a high-resolution output image. This operation benefits image super-resolution by generating a high-quality/resolution image from a low-resolution input.

This layer is added explicitly before the RRDBNet. It is essential when the scale is set to 2, as it helps rearrange information to the channel dimension and capture more features and details from the input image, ultimately contributing to generating high-quality output images with minimal artifacts. Convolutional layers here are solely responsible for the task of feature extraction along with learning the spatial patterns present on the input image of all the domains.

The Residual-in-Residual Dense Blocks (RRDB), depicted in Fig. 3, process and enhance features from the input image by integrating multilevel residual learning with dense network connections. Each RRDB block consists of several densely interconnected convolutional layers that capture considerable local features, thus boosting the network's capacity and expressiveness. Following each convolutional layer, its output merges with the initial input of the block and is fed into the subsequent layer. This structure ensures continuous connections from the previous RRDB state to all layers in the current block, creating a seamless memory mechanism that supports efficient gradient

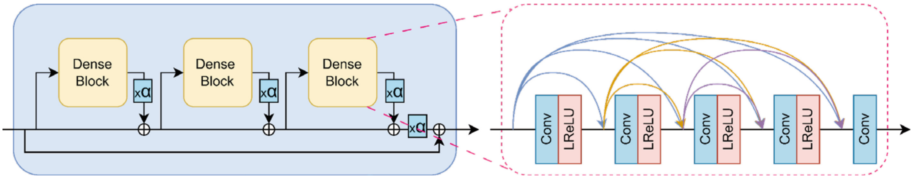
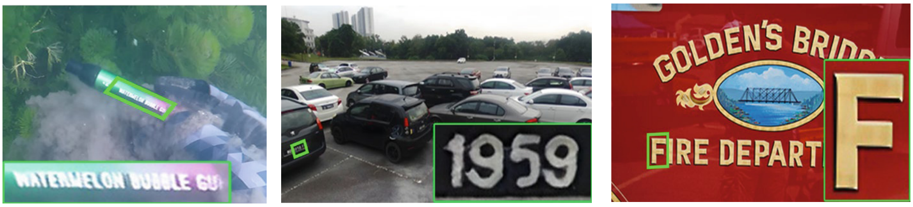


Fig. 3. Detailed architecture of a Residual-in-Residual Dense Block (RRDB) where α is the residual scaling parameter.

flow and information transfer. The RRDB blocks significantly enhance the perceptual quality of super-resolved images by producing more defined edges and lifelike textures, making them particularly effective for improving text clarity in images. The advantages of RRDB block are: i) RRDB block can extract abundant local features via densely connected convolutional layers, which increase the network capacity and expressiveness. ii) RRDB block allows direct connections from the state of the preceding RRDB to all the layers of the current RRDB, leading to a contiguous memory mechanism that facilitates the information flow and gradient propagation. iii) RRDB block helps to improve the perceptual quality of super-resolved images by producing sharper edges and more realistic textures, as shown in the experiments.



Under water Drone Scene image

Fig. 4. The effectiveness of the enhancement unit for the degraded region in the underwater, drone and scene images.

In the enhancement unit, upsampling layers use a kernel to expand the spatial resolution of the input image, ensuring key features are retained. The network achieves upsampling through the use of pixel unshuffle and convolutional layers. Sinc filters are employed to generate ringing and overshoot artifacts in training pairs. These filters are utilized during both the blurring stage and the final phase of synthesis. To enhance the discriminator's effectiveness and stabilize the training process, the network incorporates a U-Net discriminator with spectral normalization. This discriminator is responsible for differentiating real images from synthesized ones.

Thus, measuring and quantifying the gap between the predicted variables versus the actual target variables by employing adversarial loss, content loss, and perceptual loss aids the network in learning the mapping from input features to its output targets. We can derive the enhancement function to be applied on an image I as defined in Eq. (1).

$$E = F_{enhance}(I) \tag{1}$$

Here, $F_{enhance}$ includes operations of pixel unshuffle, convolution, Residual-in-Residual Dense Block processing, and upsampling which are integral parts of the enhancement unit architecture. The final output of the enhancement unit, E, is passed down to the text spotting unit. The effectiveness of the enhancement module is illustrated in Fig. 4, where it is noted that text in the underwater and drone images is significantly enhanced when compared to scene image.

3.2 Text Spotting Unit

The input image is run through the ResNet-50 network, chosen as the backbone due to its residual connections that enable deeper networks and better feature propagation, leading to improved performance even with small datasets. Moreover, ResNet-50 has fewer parameters than other state-of-the-art transfer learning methods like VGG16, making it less prone to overfitting and more computationally efficient, which is particularly beneficial when working with limited data. Unlike the box proposal adopted in previous works, from the perspective of the text center line, a straightforward Bezier center curve proposal scheme is employed. Utilizing the image features the encoder provides, a 3-layer MLP predicts offsets at each pixel of the feature maps to four Bezier control points. This setup defines a curve that represents a single text instance as shown in Fig. 5, where we can see Bezier curve finds points for fixing the tight bounding box for any oriented text lines. The top-K scored curves are selected as the proposal.

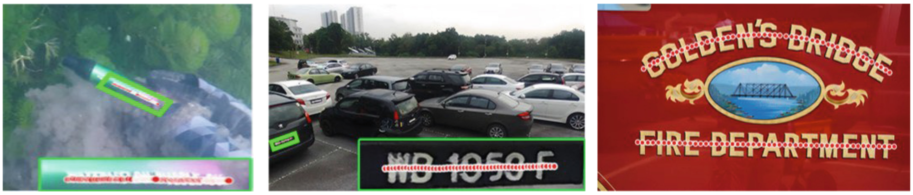


Fig. 5. Visualization of sampled on-curve points on images from Underwater, Drone and Natural Scene domains.

A set of control points generates Bezier points that define a curve formed by the collection of control points through weight summation, with weights given by Bernstein polynomials. Sample points from a defined parametric curve using a set of control points generate Bezier points. The mathematical equation for a Bezier curve of degree n is defined as in Eq. (2)

$$P(t) = \sum_{i=0}^{n} B_i J_{n,i}(t) \tag{2}$$

where $P(t)$ is the point on the curve at parameter t, B_i is the i-th control point, and $J_{n,i}(t)$ is the i-th Bernstein polynomial of degree n, is defined in Eq. (3).

$$J_{n,i}(t) = C(n, i)t^i(1-t)^{n-i} \tag{3}$$

where $C(n, i)$ is the binomial coefficient, a cubic Bezier curve has four control points as defined in Eq. (4).

$$P(t) = B_0(1-t)^3 + B_1 3t(1-t)^2 + B_2 3t^2(1-t) + B_3 t^3 \tag{4}$$

The sample points are taken from the Bezier curve and used as explicit point queries to encode the position, shape, and semantics of text. This representation simplifies the text spotting pipeline and improves the performance and efficiency of the model.

After being flattened, the multi-scale features undergo the positional embedding layer. This step helps to maintain fluidity and engagement in the sequence. Each curve proposal, defined by four Bezier control points, has N points uniformly sampled from its center, top, and bottom curves to serve as ground truth. These points' coordinates are transformed into positional queries, combined with learnable content queries to create composite queries. These are then input into a single Transformer decoder that utilizes deformable cross-attention to extract relevant text features. After passing through the decoder, the point queries embody essential text semantics and locations. This enables further decoding into the central line, boundary, center curve points, and text confidence through four straightforward, parallel prediction heads. Four simple parallel heads are adopted, each responsible for solving a specific task. The instance head predicts the instance class and boundary points. The character head predicts the character's class and center points. The center curve point head predicts the coordinate offsets from reference points to ground truth. The confidence head predicts the confidence score of the text.

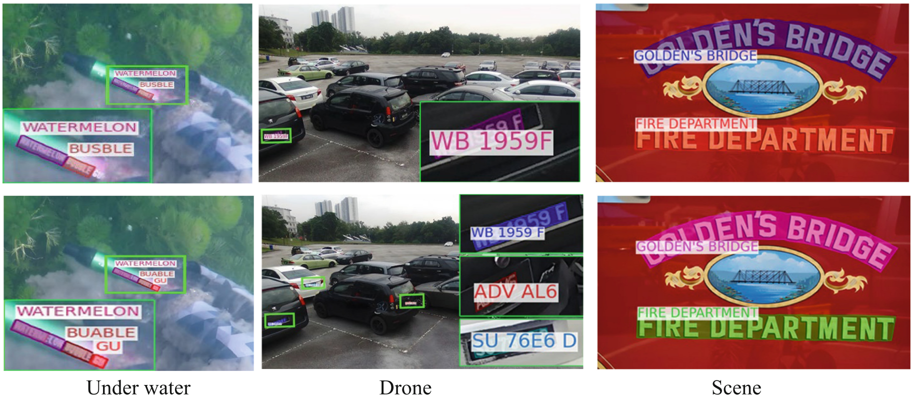


Fig. 6. The effectiveness of our text spotting. The images in the first row indicate result images of text spotting without enhancement (DeepSolo alone), while the second row indicates the result images of the proposed method after enhancement.

After the enhanced component is passed down to the spotting unit, the final output of the spotting unit can be represented by the Eqs. (5).

$$T = S(E) \tag{5}$$

Here, T is the final output of the entire proposed architecture. The function S is derived from all the operations being performed in the text spotting unit.

The spotting function, S can be defined as in Eq. (6).

$$S(E) = H(De(PE(Be(R(E))))) \tag{6}$$

In the above equation, $R(E)$ denotes the extracted features of the enhanced image using the feature extraction backbone. The operation Be represents the extraction of Bezier points from the features. The PE function denotes the positional embedding applied to the multi-scale features. De represents the processing through a single Transformer decoder. Finally, H is the decoding step to extract the center line, boundary, center curve points, and confidence of text.

The effectiveness of the text spotting is presented in Fig. 6, where it is noted that the text spotting improves significantly after enhancement (second row) compared to the before enhancement (first row), especially for underwater and drone images. However, there is no change for scene images because text spotting results are the same for both before and after enhancement. Figure 6 also infers that before image enhancement (the results of the first row), the DeepSolo alone is insufficient to tackle the challenges of underwater and drone images. At the same time, the results after enhancement (the second row) infer the combination of Real-ESRGAN and DeepSolo is capable. It is evident from the results in the second row of Fig. 6 that the number of text instances detected by the proposed model is greater than the total number of text instances detected by the DeepSolo alone. This concludes that the combination is essential to achieve the best results for images of different domains.

4 Experimental Results

To evaluate the performance of the proposed method, we consider three domains, namely, underwater, drone and scene images.

4.1 Datasets of Multiple Domains

Natural Scene Domain: Scene Domain is represented by the CTW1500 [26], ICDAR2015 [27], and Total-Text [28] datasets for the purpose of our experimentations, each offering unique challenges and features. The CTW1500 dataset, comprising 1,500 images with a split of 1,000 for training and 500 for testing. ICDAR2015, another pivotal dataset in this domain, consists of 1,500 images captured from wearable cameras, featuring around 2,077 cropped text instances annotated at different levels. Finally, the Total-Text dataset, with its 1,555 images featuring horizontal, multi-oriented, and curved text.

Underwater Domain: The Underwater domain presents unique challenges for text spotting, created at the Indian Statistical Institute, Kolkata. This domain comprises 260 word-based annotated images, with 200 designated for training and 60 for testing. The challenges of this domain are deterioration of image quality, small objects etc.

Drone Domain: In the Drone domain, the challenges associated with text spotting are distinct and are encapsulated in the domain. This domain comprises a total of 500 line-based annotated images, with 400 allocated for training and 100 for testing. The emphasis here is on scenarios encountered in aerial imaging, with considerations for crowded cars, multiple license plates, distortions, and changes in the angle of view resulting from variations in the drone's altitude.

We use standard measures for evaluating the performance of the proposed method on image enhancement, detection and end-to-end recognition (spotting). For enhancement, BRISQUE measure and for detection, measures like Precision, Recall and F1-score are used. For recognition, None, Full, S, W and G are considered. A lower BRISQUE score is indicative of higher image quality, meaning that the best-quality images receive the lowest scores within this evaluation framework.

Implementation: For enhancement, we have trained the enhancement unit with DIV2K [29], Flickr2K [30], and OutdoorSceneTraining [31] datasets. HR patch size for training is 265, with Adam as the optimizer. We train it for 400K iterations with a learning rate of 2×10^{-4}. The training process is performed on one RTX 2080Ti with batch size one. The spotting unit is pretrained on Synth150K [32], Total-Text [28], MLT17 [33], IC13 [34] and IC15 [35] datasets. We have fine-tuned the combined model (i.e., enhancement + spotting unit) on RTX 2080Ti (with 12GB VRAM) and 16GB memory for 7 h with 12000 iterations on all the discussed domains.

4.2 Ablation Study

In this work, there are two key steps, namely, image enhancement using Real-ESRGAN and text spotting using DeepSolo models. To validate how effective the enhancement is, we conducted detection and spotting experiments before enhancement and after enhancement on all three domains. We can note from the results reported in Table 1 that there is a significant improvement for underwater and drone domains with and without enhancement in terms of detection and spotting. Therefore, one can infer that the proposed enhancement is necessary and contributes immensely to degraded and poor-quality images. However, interestingly, for the scene domain, the results before and after enhancement are almost similar for both detection and spotting. This indicates that the proposed enhancement step enhances only low-contrast and poor-quality images without affecting high-quality images. This makes sense because the scene images are good quality images compared to underwater and drone images.

As mentioned in the proposed methodology section, determining patch size is also one of the parameters to improve the performance of the proposed method. To obtain optimal patch size for the different domains, we conducted experiments on the various HR patch sizes for the enhancement module, such as 128, 144, 192, 224, 256, and 265, and the results are reported in Table 2. The images of underwater and drone domains contain only one dataset, while the images of the scene text domain contain three datasets. Therefore, training and tunning of the proposed method are as good as training on individual datasets except for the scene text domain. Consequently, the same patch size for underwater and drone datasets is expected during a different patch size for the scene text domain. This is evident from the results reported in Table 2, where the patch size of 265 is the best for all the datasets regarding detection and spotting, while for scene

Table 1. Detailed comparison of the proposed method with and without the use of enhancement module for Natural Scene domain, Underwater domain and Drone domain. “None” refers to recognition without lexicon

Domains	Detection						Spotting	
	With Enhancement			Without Enhancement			With Enhancement	Without Enhancement
	P	R	F1	P	R	F1	None	None
Scene	92.3	86.2	89.1	92.0	85.0	88.7	72.9	71.95
Underwater	88.9	71.5	79.3	86.8	50.0	63.5	68.24	30.5
Drone	93.32	72.14	81.37	86.8	42.45	51.0	77.8	47.0

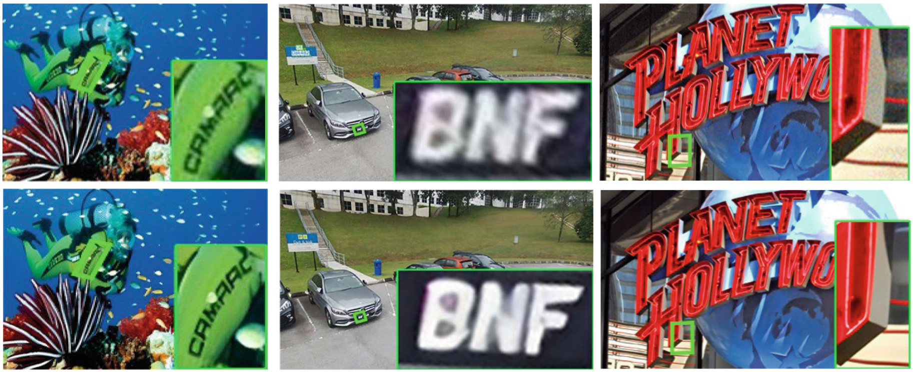
Table 2. Studying the effectiveness of different patch size for detection and spotting on datasets of different domains. “None” refers to recognition without lexicon

HR patch size	Detection									Spotting - None E2E		
	Scene			Underwater			Drone			Scene	Underwater	Drone
	P	R	F	P	R	F	P	R	F			
128	91.2	85.7	88.4	84.0	69.7	76.2	90.3	68.9	78.2	71.7	66.9	75.2
144	90.0	84.2	87.0	83.2	71.3	76.8	89.7	71.5	79.5	71.2	67.2	75.9
192	90.4	**87.7**	89.0	84.7	70.8	77.1	91.3	72.3	80.7	72.1	68.1	76.7
224	89.2	84.5	86.8	84.3	70.2	76.6	88.2	69.1	77.5	71.9	67.5	75.3
256	90.7	85.3	87.9	85.2	71.5	77.7	89.6	70.8	79.1	71.5	67.9	76.1
265	**92.3**	86.2	**89.1**	**88.9**	**71.5**	**79.3**	**93.3**	**72.1**	**81.4**	**72.88**	**68.2**	**77.8**

text, the patch size of 192 is the best at Recall. However, the patch size 265 is the best at precision and F-measure. Therefore, 265 is the optimal patch size for all our experiments.

4.3 Experiments on Enhancement

Qualitative results of the proposed method on samples of three different domains are shown in Fig. 7, where it can be seen that the text in underwater and drone images is enhanced (improved visual quality) compared to the text in input images. However, there is little change for the text in the scene image before and after enhancement. This validates the proposed enhancement works well for both poor-quality and high-quality photos. The same conclusions can be drawn from the quantitative results (BRISQUE score) reported in Table 3, where the BRISQUE score value is small compared to the value of input images for the enhanced images of the respective three domains. However, we can notice a marginal difference between the BRISQUE input score and the enhanced image for the scene domain. This justifies there is no effect of enhancement over scene images.



(a)-Underwater (b)-Drone (c)-Scene

Fig. 7. Sample qualitative results of the proposed enhancement step for three domains. The first row represents the results on input images while the second row represents the results after enhancement.

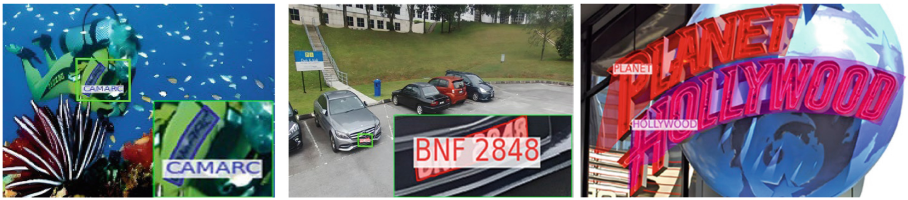
Table 3. Comparative analysis of BRISQUE scores across different domains. (Lower BRISQUE value means higher image quality)

Scene		Underwater		Drone	
Original	Enhanced	Original	Enhanced	Original	Enhanced
28.4	27.9	39.6	33.44	19.38	14.44

4.4 Experiments on Detection and Spotting

Sample qualitative results of the proposed method for text spotting can be seen in Fig. 8, where our method works well for the samples of three domains. Therefore, the proposed method can tackle poor-quality images of underwater, drone, and arbitrarily oriented/shaped text lines of scene images. While the primary aim of the proposed work is to spot text across multiple domains, we demonstrated the superiority of our method by comparing its performance to existing methods on individual datasets as well. This is because none of the existing methods reports results on three domains in the literature. A comparative study is reported in Table 4, 5 and Table 6 for respective CTW1500, Total-Text, ICDAR 2015, Underwater, and Drone datasets. For the Total-Text, the F1-score for detection and Full are higher than the existing methods.

However, the methods [3, 20] are the best at precision and recall, respectively, for detection. However, the same methods [3, 20] report poor performance in Full-spotting tasks compared to our proposed method. For the CTW1500 dataset, our method outperforms others in terms of precision, recall, and F1-score for detection, as well as overall spotting. While certain existing methods do surpass ours in detection and spotting, this is expected since they were specifically designed for scene text detection and spotting. Nevertheless, when evaluating comprehensive performance that includes both detection and spotting, our method proves to be superior to the existing approaches.



Under water Drone Scene

Fig. 8. Qualitative text spotting results of our proposed method on Underwater, Drone and Natural Scene domains.

Table 4. Detection and Spotting results of different methods on Total-Text and CTW1500 datasets. "None" refers to recognition without lexicon. "Full" refers to the usage of lexicon that includes every word present in the test set.

Methods	Detection						Spotting			
	Total-Text			CTW1500			Total-Text		CTW1500	
	P	R	F	P	R	F	Full	None	Full	None
SwinTextSpotter [18]	-	-	88.0	-	-	88.0	84.1	74.3	77.0	51.8
ABCNet v2 [19]	90.2	84.1	87.0	85.6	83.8	84.7	78.1	70.4	77.2	57.5
GLASS [20]	90.8	**85.8**	88.1	-	-	-	86.2	79.9	-	-
TESTR [21]	93.4	81.4	86.9	89.7	83.1	86.3	71.6	**83.3**	79.0	53.3
DeepSolo [3]	**93.9**	82.1	87.6	91.45	85.6	88.43	87.0	79.7	81.40	**64.20**
Proposed method	93.36	84.51	**88.72**	**92.00**	**86.38**	**89.1**	**88.05**	81.79	**81.44**	63.97

For the ICDAR 2015 dataset, our proposed method achieves the highest precision in detection and excels in the *S* for spotting, outperforming other leading methods. This indicates that our method is versatile, effectively handling both simple and complex scene datasets. In contrast, for the ICDAR2015 dataset, method [3] secures the top F-score in detection and leads in the *W* and *G* metrics for spotting. This discrepancy is likely due to the reduced complexity encountered when focusing on individual datasets rather than multiple domains. Regarding the underwater dataset, our method stands out in recall, F-score, and None metrics. Similarly, it achieves the best precision and None for the drone dataset. However, method [23] records the highest precision for underwater scenarios and [2] tops in recall and F-score for drone imagery. This is because the method [23] was developed to address the challenges of poor-quality images like underwater images through enhancement. Therefore, the scope of existing methods is limited to a particular domain but not multiple domains. Overall, when we analyze the performance of the proposed method on all the datasets, our method is superior to existing methods in terms of detection and spotting.

Table 5. Text spotting performance of the proposed and the state-of-the-art systems on the ICDAR-15 datasets. Here "S", "W", "G" represent recognition with "Strong", "Weak", "Generic", lexica, respectively

Methods	Detection			Spotting		
	P	R	F	S	W	G
ABCNet v2 [19]	90.4	86.0	88.1	82.7	78.5	73.0
TESTR [21]	90.3	**89.7**	90.0	85.2	79.4	73.6
SwinTextSpotter [18]	-	-	-	83.9	77.3	70.5
SPTS [22]	-	-	-	77.5	70.2	65.8
DeepSolo [3]	90.68	87.4	**90.0**	86.8	**81.9**	**76.9**
Proposed method	**91.41**	87.8	89.57	**87.12**	80.53	72.6

Table 6. Low quality image text spotting results on Underwater and Drone datasets. "None" refers to recognition without a lexicon.

Underwater Dataset					Drone Dataset				
Methods	P	R	F	None	Methods	P	R	F	None
Banerjee et al. [1]	90.25	45.37	60.38	-	TESTR [21]	54.0	40.6	46.4	13.1
DA-TextSpotter [23]	**95.65**	48.73	64.57	64.15	Hamam Mokayed et al. [2]	83.2	**86.2**	**83.9**	-
TESTR [21]	92.2	33.8	49.54	29.63	DeepSolo [3]	86.76	42.45	51.0	47.0
SwinTextSpotter [18]	83.2	34.4	29.0	29.08	-	-	-	-	
DeepSolo [3]	86.81	50.0	63.45	30.52	-	-	-	-	-
Proposed method	88.9	**71.5**	**79.3**	**68.24**	**Proposed method**	**93.32**	72.14	81.37	**77.8**

To test the performance of the proposed and existing methods on different domains, we calculated measures for each domain separately, and the results are reported in Table 7. We emphasize domain-based comparisons with other models rather than using individual datasets within the domain. It is observed from Table 7 that the existing models, including DeepSolo, do not perform well for underwater and drone images compared to the proposed method. Therefore, one can infer that the existing models lack generalization ability and domain independence. On the other hand, the proposed model works well for different domains and individual datasets.

When we compare the performance of the proposed and existing methods on individual datasets and domain datasets, the performance of the proposed method is consistent for the three domains compared to state-of-the-art methods. However, the performance of the proposed method is competitive and promising for the individual datasets compared to the state-of-the-art methods. The reason is as follows. The main objective of the proposed work is to develop a model that can work well for different domains irrespective of degradations caused by underwater, drone, and quality variations in scene images. At the same time, the key objective of the existing models is to achieve the best result for a particular type of images.

Table 7. Performance of the proposed and existing methods on different domains

Method	Detection									Spotting		
	Scene			Underwater			Drone			Scene	Underwater	Drone
	P	R	F1	P	R	F1	P	R	F1	None	None	None
ABCNet v2 [19]	88.7	84.6	86.6	-	-	-	-	-	-	64.0	-	-
TESTR [21]	91.1	84.7	87.7	**92.2**	33.8	49.5	54.0	40.6	46.4	68.3	29.6	13.1
DeepSolo [3]	92.0	85.0	88.7	86.8	50.0	63.5	86.8	42.45	51.0	71.95	30.5	47.0
Proposed	**92.3**	**86.2**	**89.1**	88.9	**71.5**	**79.3**	**93.3**	**72.1**	**81.4**	**72.88**	**68.2**	**77.8**

4.5 Cross Domain Validation

We conducted cross-domain experiments to validate the proposed method's domain independence. In this case, the method uses samples from different domains for training/testing. Consistent results are expected when the method uses the training and testing samples chosen from two domains. The results reported in Table 8 show that our proposed model shows consistent performance across domains. However, we can observe a deterioration in the performance of the drone dataset in the cross-validation experiments. The main reason is the inherent complexity of the dataset. The images in the drone dataset are low-resolution license plate images, with various challenges such as image distortion, higher altitude, extreme angle changes, and varying orientations of the license plates. Another important reason is that the license plate number does not provide semantic information to correct the text for deep learning models. Therefore, when we train on these samples, the models do not learn at a high level to spot text in other images, and hence mismatch may occur at semantic level for achieving consistent results for the drone domain. This is beyond the scope of the proposed work.

Although the proposed method is domain-independent, when the text is not visible and readable from our naked eyes due to tiny text blur, the proposed method fails to detect and spot the text, as shown in sample cases in Fig. 9, where our method does not detect text. The images shown in Fig. 9 are the output of the entire proposed architecture, which indicates that tiny, unreadable text in an image can go undetected by the model even after being enhanced. Therefore, this has been marked as a limitation of this work and our

Table 8. Detailed cross-domain validation results where (i) and (ii) represent the results of our proposed method trained and tested online-based annotated domain, whereas (iii) and (iv) represent the results of our proposed method trained and tested on word-based annotated domains.

#	Cross-Domains Validation	P	R	F1	None
(i)	Drone → Natural Scene (CTW1500)	94.2	82.85	88.16	64.37
(ii)	Natural Scene (CTW1500) → Drone	89.84	41.37	56.67	19.00
(iii)	Natural Scene (TotalText+ICDAR2015) → Underwater	96.59	72.8	82.57	55.89
(iv)	Underwater → Natural Scene (TotalText + ICDAR2015)	92.32	77.08	83.94	72.46

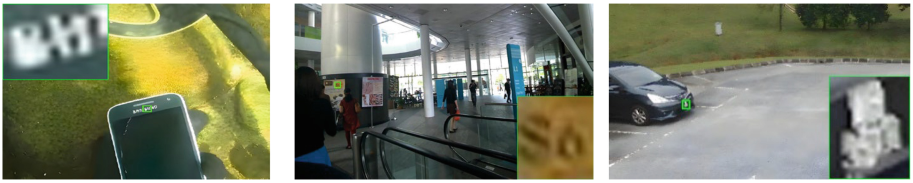


Fig. 9. Examples of images where text is barely visible in all the three cases.

future work. To overcome this problem, we plan to implement feedback mechanisms to improve the visual quality of the images through an end-to-end approach. This is our future work.

5 Conclusion and Future Work

We have proposed a new model for text spotting in multiple domains, namely, underwater, drone, and scene images. Unlike most existing methods that focus on natural scene images or particular types of datasets, the proposed work focuses on multiple domains; each domain has its challenges. To address such complex problems, we have explored the combination of Real-ESRGAN and DeepSolo, which integrate enhancement and spotting in a novel way to make the model domain independent. We have done several experiments, such as ablation study, calculating quality measures, detection, spotting, and cross-domain validation, to show that the proposed method is superior to existing methods. However, as discussed in the experimental section, the proposed method does not work well when the images consist of tiny text and lose visibility due to blur and other distortion. To overcome this challenge, one can think of a feedback mechanism based on the quality of the images through an end-to-end approach. This will be our future target.

Acknowledgement. This work is partially supported by IDEAS-TIH, ISI Kolkata.

References

1. Banerjee, A., Shivakumara, P., Pal, S., Pal, U., Liu, C.L.: DCT-DWT-FFT based method for text detection in underwater images. In: Proceedings of the ACPR, pp. 218–233 (2022)
2. Mokayed, H., Shivakumara, P., Woon, H.H., Kankanhalli, M., Lu, T., Pal, U.: A new DCT-PCM method for license plate number detection in drone images. Pattern Recogn. Lett. **148**, 45–53 (2021)
3. Maoyuan, Y., et al.: Deepsolo: let transformer decoder with explicit points solo for text spotting. In: Proceedings of the CVPR, pp. 19348–19357 (2023)
4. Xintao, W., Xie, L., Dong, C., Shan, Y.: Real-ESRGAN: training real-world blind super-resolution with pure synthetic data. In: Proceedings of the ICCV, pp. 1905–1914 (2021)
5. Jianqi, M., Guo, S., Zhang, L.: Text prior guided scene text image super-resolution. IEEE Trans. IP 1341–1353 (2023)
6. Minglong, X., Huang, Z., Liu, R.-Z., Lu, T.: A novel attention enhanced residual-in-residual dense network for text image super-resolution, pp. 1–6 (2021)
7. Shivakumara, P., Banerjee, A., Pal, U., Nandanwar, L., Lu, T., Liu, C.-L.: A new language-independent deep CNN for scene text detection and style transfer in social media images. IEEE Trans. IP (2023)
8. Yuliang, L., et al.: SPTS v2: single-point scene text spotting. arXiv preprint arXiv:2301.01635 (2023)
9. Banerjee, A., Shivakumara, P., Bhattacharya, S., Pal, U., Liu, C.L.: An end-to-end model for multi-view scene text recognition. Pattern Recogn. **149x**, 110206 (2024)
10. Jianqi, M., Liang, Z., Zhang, L.: A text attention network for spatial deformation robust scene text image super-resolution. In: Proceedings of the CVPR, pp. 5911–5920 (2022)
11. Shancheng, F., Mao, Z., Xie, H., Wang, Y., Yan, C., Zhang, Y.: Abinet++: autonomous, bidirectional and iterative language modeling for scene text spotting. IEEE PAMI (2022)
12. Xixuan, H., Zhang, A., Meng, X., Fu, B.: Deformation robust text spotting with geometric prior. In: Proceedings of the ICIP, pp. 3573–3577 (2023)
13. Zepeng, H., Wan, Q., Chen, J., Zhao, X., Ye, K., Shen, L.: ADATS: adaptive RoI-align based transformer for end-to-end text spotting, pp. 1403–1408 (2023)
14. Taeho, K., Kim, S., Seo, S., Kim, Y., Kim, D.: Towards unified scene text spotting based on sequence generation. In: Proceedings of the CVPR, pp. 15223–15232 (2023)
15. Weijia, W., et al.: DSText V2: a comprehensive video text spotting dataset for dense and small text. Pattern Recogn. **149**, 110177 (2024)
16. Nicolas, C., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Proceedings of the ECCV, pp. 213–229 (2020)
17. Yuzhong, Z., Wu, W., Li, Z., Li, J., Wang, W.: FlowText: synthesizing realistic scene text video with optical flow estimation. arXiv preprint arXiv:2305.03327 (2023)
18. Mingxin, H., et al.: Swintextspotter: scene text spotting via better synergy between text detection and text recognition. In: Proceedings of the CVPR, pp. 4593–4603 (2022)
19. Yuliang, L., et al.: Abcnet v2: adaptive Bezier-curve network for real-time end-to-end text spotting. IEEE PAMI 8048–8064 (2021)
20. Roi, R., Tsiper, S., Anschel, O., Lavi, I., Markovitz, A., Manmatha, R.: Glass: global to local attention for scene-text spotting. In: Proceedings of the ECCV, pp. 249–266 (2022)
21. Xiang, Z., Su, Y., Tripathi, S., Tu, Z.: Text spotting transformers. In: Proceedings of the CVPR, pp. 9519–9528 (2022)
22. Dezhi, P., et al.: SPTS: single-point text spotting. In: Proceedings of the ACMMM, pp. 4272–4281 (2022)
23. Das, A., Biswas, S., Pal, U., Llados, J.: Diving into the depths of spotting text in multi-domain noisy scenes (2023). https://doi.org/10.48550/arXiv.2310.00558

24. Mokayed, H., Shivakumara, P., Alkhaled, L., Al-Masr, A.N.: License plate number detection in drone images. In: Artificial Intelligence and Applications, pp 1–8 (2022)
25. Pal, S., Roy, A., Shivakumara, P., Pal, U.: Adapting a swim transformer for license plate number detection and text detection in drone images. In: Artificial Intelligence and Applications, pp 145–154 (2023)
26. Liu, Y.: Detecting curve text in the wild: new dataset and new solution. arXiv. Published online December 6, 2017. https://doi.org/10.48550/arXiv.1712.02170
27. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., et al.: ICDAR 2015 competition on robust reading. In: Proceedings of the ICDAR, pp. 1156–1160 (2015). https://doi.org/10.1109/ICDAR.2015.7333942
28. Ch'ng, C.K., Chan, C.S.: Total-Text dataset. Published online 2017. https://github.com/cs-chan/Total-Text-Dataset. Accessed 14 June 2023
29. Agustsson, E., Timofte, R.: NTIRE 2017 challenge on single image super-resolution: dataset and study. In: Proceedings of the CVPRW, pp. 1122–1131 (2023)
30. Timofte, R., Agustsson, E., Van Gool, L., et al.: NTIRE 2017 challenge on single image super-resolution: methods and results. In: Proceedings of the CVPRW, pp. 1110–1121 (2017)
31. Feng, W., Guo, Z., Zhang, Z., Zhang, W.: OutdoorSceneTraining (OST). Baidu (2015)
32. Liu, Y., Chen, H., Shen, C., He, T., Jin, L., Wang, L.: Abcnet: real-time scene text spotting with adaptive Bezier-curve network. In: CVPR (2020)
33. Nayef, N., Patel, Y., Busta, M., et al.: ICDAR2017 robust reading challenge on multi-lingual scene text detection and script identification - RRC-MLT. In: Proceedings of the ICDAR (2017)
34. Karatzas, D., et al.: ICDAR 2013 robust reading competition. In: ICDAR (2013)
35. Karatzas, D., et al.: ICDAR 2015 competition on robust reading. In: ICDAR (2015)

LineTR: Unified Text Line Segmentation for Challenging Palm Leaf Manuscripts

Vaibhav Agrawal, Niharika Vadlamudi, Muhammad Waseem, Amal Joseph, Sreenya Chitluri, and Ravi Kiran Sarvadevabhatla(✉)

Centre for Visual Information Technology, International Institute of Information Technology, Hyderabad, Hyderabad 500032, India
ravi.kiran@iiit.ac.in

Abstract. The dense and unstructured text in historical manuscripts presents significant challenges for precise line segmentation due to large diversity in sizes, scripts and appearances of the documents. Existing approaches tackle this complexity either by performing dataset-specific processing or training per-dataset models. This strategy hampers maintainability and scalability as newer manuscript collections get digitized and annotated. In this paper, we propose **LineTR**, a novel two-stage line segmentation approach which can process a diverse variety of challenging handwritten documents in a unified, dataset-agnostic manner. **LineTR**'s first stage processes context-adaptive image patches. It consists of a novel DETR-style network which generates parametric representations of text strike-through lines (scribbles) and a novel hybrid CNN-transformer network which generates a text energy map. A dataset-agnostic and robust post-processing procedure is applied on first-stage outputs to obtain document-level scribbles. In the second stage, these scribbles and the text energy map are used within a seam generation framework to obtain highly precise polygons enclosing the manuscript text lines. We also introduce three new diverse text line segmentation datasets comprising challenging Indic and South-East Asian manuscripts. Through experiments, ablations and evaluations, we show that **LineTR** generates significantly superior line segmentations - *all with a single model*. Our results also highlight the effectiveness of our unified model for good quality zero-shot inference on the newly introduced datasets. Project page: https://ihdia.iiit.ac.in/LineTR/.

Keywords: Text Line Segmentation · Historical Manuscripts · Deep Learning · Zero-Shot · Transformers

1 Introduction

Many approaches have been proposed in recent years to improve the quality of text line segmentation in challenging historical manuscripts [6,21,27,30,31]. Despite their successes, fundamental challenges remain. For instance, manuscript attributes such as size, aspect ratio, text line density, script, diacritics often

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 217–233, 2025.
https://doi.org/10.1007/978-3-031-78495-8_14

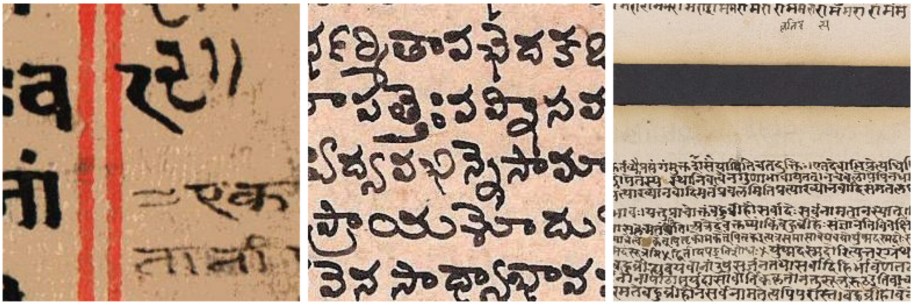
change dramatically across datasets. For existing approaches, this diversity of attributes is not easy to handle in a unified manner. As a compromise, dataset-specific hyperparameters are used in some cases. In other cases, dataset-specific models are trained. However, as new manuscript collections get digitized and annotated, the current strategies are not practical from maintenance and scalability point of view. Consequently, there is a practical need for an approach which can process a diverse variety of manuscripts in a unified, dataset-agnostic manner while delivering good line segmentation performance.

Motivated by this need, we propose **LineTR**, a unified, highly adaptive and precise text segmentation approach for challenging historical manuscripts. **LineTR** is designed as a two-stage pipeline [31]. To begin with, overlapping patches are sampled from input image in a context-adaptive manner. The first stage consists of [i] a novel DETR-style [8] deep network which generates parametric representations of text strike-through lines (scribbles) and [ii] a novel hybrid CNN-transformer network which generates a text energy map. A dataset-agnostic and robust post-processing procedure is applied on patch-level predictions from the first stage to obtain document-level scribbles. The scribbles and text energy map are used within an enhanced seam generation framework (the second stage) to obtain highly precise polygons enclosing the manuscript text lines.

Our first insight is that documents occur in various resolutions and aspect ratios, and resizing them to a fixed size causes problems. If the patch size is too large, then the number of learnable parameters as well as the memory and time complexity increase dramatically, and otherwise if the patch size is too small, then a lot of information in the image, such as the separation between the text lines is gone. This motivates a patch-based approach, similar to [31]. However, SeamFormer [31] samples patches of a fixed size (256×256), which brings in the issue of *bad context*. A patch of a fixed size can contain vastly varying amounts of information or *context*, depending on the resolution of the original document (see Fig. 1), which is not desirable for a system which works generally across datasets. Therefore, an *adaptive* patching mechanism is required which can find the right patch size for a given document. We have shown in our experiments that this is a crucial design decision.

Our second insight is that a text line is a geometric structure (a curve), and this can serve as an excellent prior for the task of text-line detection [9, 11, 15, 22, 23, 35]. Segmentation based approaches [5, 7, 16, 17, 19, 20, 25, 27, 31] do not utilize this prior, and instead predict a *blob* of pixels as the representation of a text-line, which leads to unwanted artifacts such as merging of adjacent blobs (see Fig. 2). We approximate a text line within a patch using a straight line (see Fig. 3), and this approach has clear benefits over segmentation based approaches, as indicated by our results.

Finally, we also introduce three new diverse text line segmentation datasets consisting of challenging Indic and South-East Asian manuscripts. Through experiments, ablations and evaluations on new and existing palm leaf manuscript datasets, we show that **LineTR** generates significantly superior line segmenta-



(a) too *little* context!

(b) *good* context

(c) too *much* context!

Fig. 1. Fixed size patches (256 × 256, as used in [31]) sampled from different documents can result in *poor context*.

tions compared to other competing approaches (Sect. 7) - all with a single unified model. Additionally, we highlight the effectiveness of our unified model for good quality zero-shot inference on the newly introduced datasets (Sect. 7). Additional diagrams, zero-shot results and other details can be found on the project page.

2 Related Work

For an overview of methods ranging from classical image processing to deep learning techniques, refer to Vadlamudi et al. [31]. In some approaches, the document is fed to a neural network which directly predicts the text line polygonal instances or regions [5,7,12,16,17,19,20,24,25,27]. These one-shot approaches typically involve downsampling the image to a fixed input dimension. Therefore, they do not work well for high-aspect ratio palm leaf manuscripts containing closely spaced lines. Despite promising results on other historical manuscripts, these approaches predict imprecise text-line polygons and exclude vital textual components (e.g. diacritics).

A popular alternative is to predict 1-D geometric structures related to the text line. These structures are often represented as underlines (baselines) [9,11,22] or strike-throughs [23,35]. Instead of a pixel-based representation, Kiessling et al. [15] propose an approach which predicts the Bezier coefficients of the baseline.

While some approaches treat the gap between adjacent baselines as text lines [2], others employ heuristics to obtain text-line polygons [26]. Since these approaches also involve input downsampling, they inherit the shortcomings of one-shot approaches mentioned earlier. In contrast, our approach for **LineTR** does not involve manuscript image downsampling.

Another popular line of research employs seam generation techniques for delineating text-line polygons or combining seams to form text line polygons [1]. These methods employ the conventional seam carving algorithm [4] on

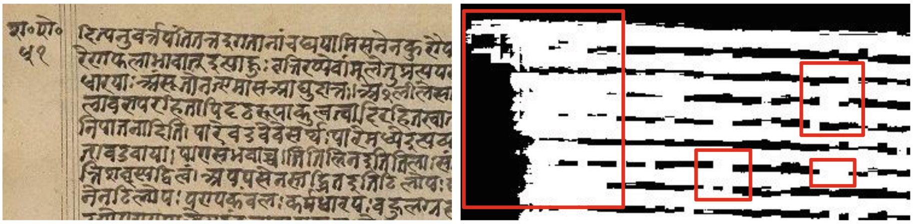


Fig. 2. Issues with SeamFormer's [31] raw scribble map output on a dense Indic manuscript [27]. SeamFormer formulates scribble prediction as a per-pixel binary classification task, leading to extremely noisy predictions. Highlighted regions in the image cannot be post-processed to obtain distinct scribbles because of extreme merging.

curated energy maps to detect baselines or to generate separators for text-line regions. The novelty in these approaches often originates from the proposed energy map. This is exemplified by Signed Distance Transform (SDT) for Arabic manuscripts [26], geodesic distance transform energy map [3], a global energy map that incorporates diacritics [31]. These methodologies often rely on classical image processing techniques. Vadlamudi et al. [31] propose a hybrid two-stage approach in which strike-through regions predicted by a first stage network are used along with multiple energy maps to generate medial, upper and lower seams associated with the line polygon. While these approaches show promise, they necessitate intensive hyper-parameter tuning for the energy maps for every new dataset. While our approach uses seam generation, we introduce a hybrid CNN-transformer module which produces a single energy map and generalizes across diverse document styles and languages.

To avoid the drawbacks associated with image downsampling, some approaches partition the image into smaller patches and predict the text-line fragments [6]. The fragments are typically merged into document-level polygons using heuristic post-processing methods [31] or seam generation [6]. However, the post-processing is fragile and not viable when the lines have dense and uneven geometry as found in palm leaf manuscripts. Although SeamFormer [31] reduces the fragility by predicting text strike-through scribbles at patch level, the reliance on dataset-specific post-processing is not fully resolved due to noisy pixel-based representation of the scribbles. Often, the predictions merge into each other, as shown in Fig. 2. In contrast, we employ a parametric line representation for scribbles which enables dataset-agnostic post-processing. Unlike existing works which use fixed size patches, **LineTR** uses variable-sized patches which aids generalization.

As a rule, existing approaches train separate models and employ heuristics which are dataset-specific [6,31,35]. As emphasized earlier, ours is the first approach for text-line segmentation which works across multiple diverse datasets without requiring dataset-specific processing. As we have shown, this ability also enables **LineTR** to exhibit good zero-shot performance on unseen datasets.

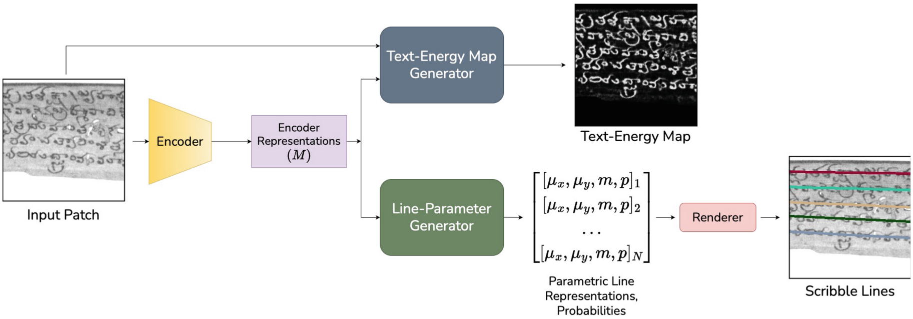


Fig. 3. Stage-1 pipeline (Sect. 3.1). Scribbles lines are overlaid on the input patch for clarity.

3 The Proposed Approach: LineTR

We adopt a two stage approach to predict text line polygons. In *Stage-1* (see Fig. 3), the input image is first split into overlapping contextual patches of various sizes (Sect. 3.4.1). Each patch is processed by a deep network which predicts (a) parametric representations of text strike-through lines (scribbles) and (b) a continuous binary energy map (Sect. 3.1). The per-patch outputs are merged using an adaptive, data-agnostic post-processing module to obtain a global scribble map and a global binary energy map (Sect. 3.4.2). *Stage-2:* The global maps from *Stage-1* are processed using a seam generation algorithm to obtain tight-fitting polygons enclosing the text lines in the image (Sect. 3.5).

3.1 Stage-1

In this stage, the input patch is first processed by a shared encoder (see Fig. 3). The encoder's representations (M in Fig. 3, 4) are fed to the Line-Parameter Generator which outputs parametric representations of scribble segments in the patch. The encoder representations are also fed to the Text-Energy Map Generator which outputs a continuous ($[0, 1]$) binary energy map. Next, we describe the individual components of *Stage-1* pipeline.

3.1.1 The Encoder

This is comprised of a Vision Transformer (ViT) [10] which outputs feature map M – the encoder representation.

3.2 Line-Parameter Generator

3.2.1 Scribble Representation

We represent each scribble line using three parameters (Fig. 4)– μ_x, μ_y and m where (μ_x, μ_y) represents the mid-point of the scribble segment and m represents the slope (see Fig. 5). μ_x and μ_y are normalized wrt patch dimensions by dividing with image width and height respectively so that $\mu_x, \mu_y \in [0, 1]$.

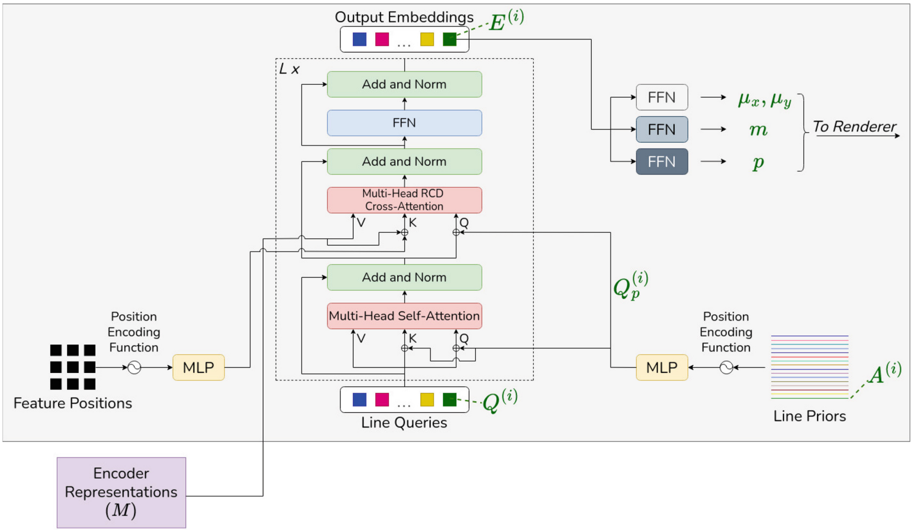


Fig. 4. Line-Parameter Generator

3.2.2 Architectural Details

We introduce a novel DETR-style [8] framework to predict parametric representations for each scribble line. A set of N learnable embeddings, which we call 'Line Queries' are fed to a transformer decoder (see Fig. 4). Within the decoder, these line queries are processed along with encoder representations M to obtain latent representations for the scribble lines ('Output Embeddings' in Fig. 4). These latent representations are transformed via lightweight feed-forward networks (FFN) to obtain scribble line parameters μ_x, μ_y, m and associated probabilities p (see top-right in Fig. 4). Next, we describe some key components of the transformer decoder.

Line Queries: We first define 'Line Priors'. These are N horizontal lines distributed uniformly throughout the image (see bottom right corner of Fig. 4). Formally, the i-th line prior $A^{(i)}$ is parameterized as $\mu_x = 0.5, \mu_y = i/N, m = 0$. We define positional query $Q_p^{(i)}$ as the positionally encoded and transformed version of $A^{(i)}$. Each line query $Q^{(i)}$ is first initialized to zero. We add the positional query $Q_p^{(i)}$ at the inputs of the attention layers (see Fig. 4).

Self Attention: There are two types of attention in the transformer decoder – self attention and cross attention [33]. The self attention is applied within the *Line Queries* Q (defined previously). Note that the positional queries are added to the line queries before computing the attention scores.

Cross Attention: The outputs of *Self Attention* are fed to the cross attention module. Due to the large size of the encoder feature representation M, the number of attention weights is quite large, which leads to slow convergence in DETR-style frameworks [8,34,37]. To counter this, we adopt a decoupled

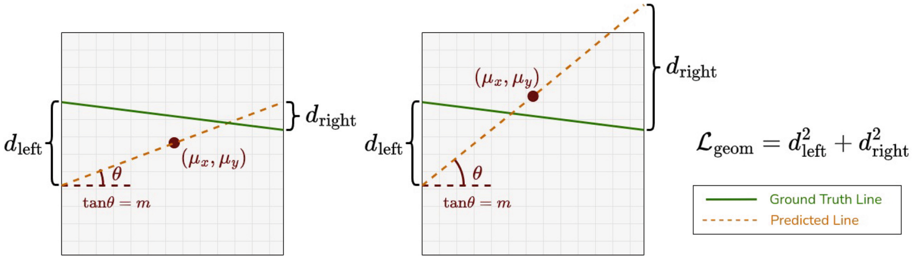


Fig. 5. Our proposed loss function for penalizing the misalignment between two lines. If the scribble line touches the top or the bottom patch boundary, then we use interpolation to calculate the loss as shown in the figure on the right.

row-column-based attention proposed by Wang et al. [34] which leads to faster convergence.

Final Predictions: The line queries undergo successive layers of self-attention and cross-attention with the encoder representations. Ultimately, each query $Q^{(i)}$ is transformed into an output embedding $E^{(i)}$ - see Fig. 4. Each $E^{(i)}$ either represents a scribble or ϕ (the empty class). We obtain the scribble-line parameters μ_x, μ_y, m along with line probability score p for each embedding using feed-forward networks (see Fig. 4).

3.2.3 Optimization

Having obtained the predicted parameters and line probability scores for each query, we match each ground truth line to a query such that the assignment is one-to-one and optimal using the Hungarian Algorithm.

Let $\{l_i\}_{i=1}^{N_0}$ be the set of ground truth lines sorted by their μ_y values. We define the median vertical gap δ between the sorted lines as $\delta = \texttt{median}\{\mu_{y_2} - \mu_{y_1}, \mu_{y_3} - \mu_{y_2}, ..., \mu_{y_{N_0}} - \mu_{y_{N_0-1}}\}$. Let $\hat{l}(\hat{\mu_x}, \hat{\mu_y}, \hat{m})$ be a predicted line with associated probability p and l (μ_x, μ_y, m) be a ground truth line. We propose a geometry-based loss function for penalizing the misalignment between the predicted line $\hat{l}$ and the ground truth line l. Let d_{left} and d_{right} be the vertical distances between the lines at the left and the right patch boundary respectively (see Fig. 5). We define the geometric loss $\mathcal{L}_{\text{geom}}$ as: $\mathcal{L}_{\text{geom}}(\hat{l},\ \text{l}) = d^2_{\text{left}} + d^2_{\text{right}}$.

Finally, we define the matching cost function $\mathcal{C}_{\text{match}}$ as follows:

$$\mathcal{C}_{\text{match}}(\hat{l}, l) = \lambda_{\text{geom}} \cdot \frac{\mathcal{L}_{\text{geom}}(\hat{l}, l)}{\delta} - \lambda_p \cdot p$$

where $\lambda_{\text{geom}}, \lambda_p \in \mathbb{R}$ are hyperparameters. After the matching, some queries would be assigned to a line. The rest of the queries would be assigned ϕ (the empty class). Let $g : \mathbb{Z}^+ \rightarrow \{\mathbb{Z}^+, \phi\}$ be the obtained matching, such that $g(i) = j$ if the i^{th} prediction is matched to the j^{th} ground truth line, and $g(i) = \phi$ if it is assigned ϕ. We define the optimization loss $\mathcal{L}_{\text{opt}}$ as follows:

$$\begin{aligned}
\mathcal{L}_{\text{opt}} &= \mathcal{L}_1 + \mathcal{L}_2 \\
\mathcal{L}_1 &= \sum_{i=1;g(i)=\phi}^{N} \left[\lambda_1 \mathcal{L}_{\text{focal}}(p_i, -) \right] \\
\mathcal{L}_2 &= \sum_{i=1;g(i)\neq\phi}^{N} \left[\lambda_2 \mathcal{L}_{\text{focal}}(p_i, +) + \lambda_{\text{geom}} \frac{\mathcal{L}_{\text{geom}}(\hat{l}, l)}{\delta^2} \right]
\end{aligned}$$

where $\lambda_{\text{geom}}, \lambda_1, \lambda_2 \in \mathbb{R}$ are hyperparameters, and $\mathcal{L}_{\text{focal}}$ stands for the focal loss [18]:

$$\begin{aligned}
\mathcal{L}_{\text{focal}}(p_i, +) &= -(1 - p_i)^{\gamma} \log(p_i) \\
\mathcal{L}_{\text{focal}}(p_i, -) &= -(p_i)^{\gamma} \log(1 - p_i)
\end{aligned}$$

where γ is a hyperparameter. This completes the description of 'Line-Parameter Generator' module in *Stage-1*. Next, we describe the 'Text-Energy Map Generator' (see Fig. 3).

3.3 Text-Energy Map Generator

The text-energy map generator is used to generate a continuous binary ($[0, 1]$) map to be used as an input to the seam generation algorithm in *Stage-2*. It uses a hybrid CNN-transformer architecture (see Fig. 6). The input patch is fed to a CNN encoder which outputs a feature map P. This feature map and representation M from the backbone encoder are fed to a transformer decoder. Within the decoder, self attention is applied to M and the result is processed along with P via a standard cross attention mechanism [33]. The resulting output is decoded to the target binary map via a CNN decoder containing skip connections with intermediate feature maps of the CNN encoder.

For optimization, we use the focal loss [18]:

$$\mathcal{L} = -\alpha \cdot y \cdot (1 - \hat{y})^{\gamma} \log\hat{y} - (1 - \alpha) \cdot (1 - y) \cdot \hat{y}^{\gamma} \log(1 - \hat{y})$$

where $\hat{y}$ represents the sigmoid activation layer's output (shaded blue in Fig. 6), $y \in \{0, 1\}$ represents the ground truth, and α, γ are hyperparameters.

3.4 Stage-1 Inference and Post-Processing

In this section, we describe the mechanism by which global document-level strikethrough scribbles and Text-Energy map are obtained during test time (inference). The exact algorithms and visual explanations of the mechanisms can be found on the project page.

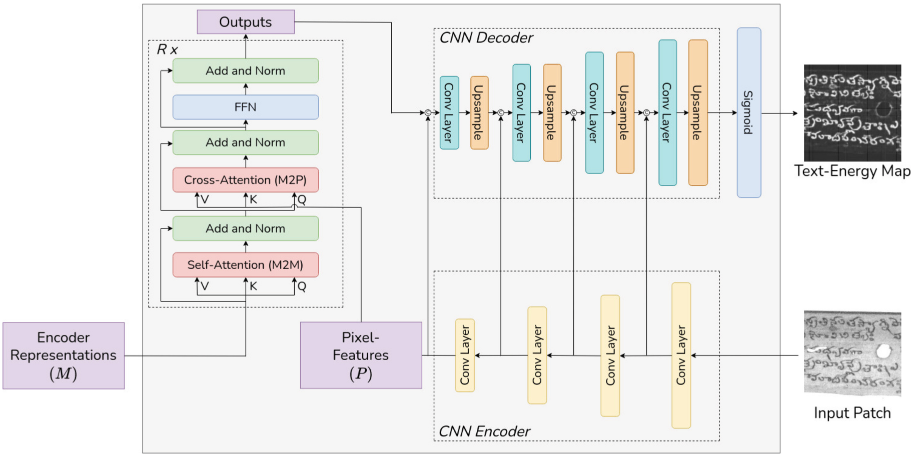


Fig. 6. Text-Energy Map Generator (Color figure online)

3.4.1 Context-Adaptive Patching

Documents occur in various sizes. To ensure that the patches contain enough contextual information for effective scribble line prediction, we introduce an adaptive patching mechanism. Given the input image, we sample patches of various sizes, and perform inference to obtain initial scribble line outputs. These are used to estimate the average spacing between text lines. A patch size t is calculated, such that the patches capture *suitable* context in the document. We then sample patches of size t and perform inference to obtain scribble-line predictions. These predictions tend to be more accurate because the patches are context-adapted. See Fig. 7 for a visual description. Refer to the project page for more details.

3.4.2 Combining Patch-Level Outputs

After obtaining context adapted patches (Sect. 3.4.1) and patch-level scribble-line predictions (Sect. 3.2), we construct the global scribble map $\mathcal{S}$ using an iterative *Projection-Merging Algorithm*. Roughly, the algorithm involves rendering each patch's line predictions on the original document, which are then clustered based on distance between them. Each cluster so formed represents a text line in the document. The exact algorithm along with a visual explanation can be found on the project page.

To obtain the global Text-Energy Outputs, we sample the original image into non-overlapping patches and pass them through the Text-Energy Map generator (Sect. 3.3) to obtain patch-level outputs. These are then combined to form the global Text-Energy map $\mathcal{B}$. See the project page for more details.

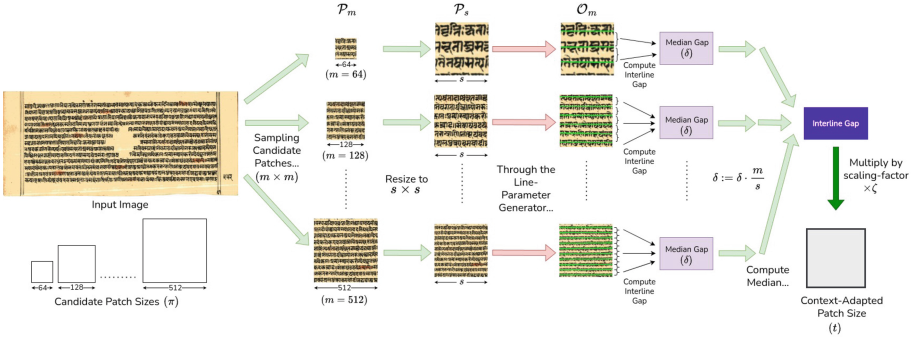

$\mathcal{P}_m$
$\mathcal{P}_s$
$\mathcal{O}_m$
Input Image
Sampling Candidate Patches... $(m \times m)$
64
$(m = 64)$
128
$(m = 128)$
512
$(m = 512)$
Candidate Patch Sizes (π)
Resize to $s \times s$
s
Through the Line-Parameter Generator...
Compute Interline Gap
Median Gap (δ)
$\delta := \delta \cdot \frac{m}{s}$
Compute Median...
Interline Gap
Multiply by scaling-factor $\times\zeta$
Context-Adapted Patch Size (t)

Fig. 7. First, *candidate* patches of various sizes $m = \{64, 128, \ldots 512\}$ are sampled from the document. Line-parameter predictions are made for each patch by resizing them to the input shape expected by the model ($s = 256$) and then passing them through the Line-Parameter Generator (Sect. 3.2). An interline-gap value δ for each patch is calculated using the line predictions for the patch. These patch-level interline gap predictions are then scaled to the size of the original document ($\delta := \delta \cdot \frac{m}{s}$) and averaged to obtain an estimate of the average interline gap value for the whole document. This averaged value is multipled by a scaling factor ζ, which roughly represents the expected number of text lines seen in a patch to obtain the final context-adapted patch size (t). Patches of size t are then sampled from the original document, and are fed to the Line-Parameter Generator to get the final predictions.

3.5 Stage-2

The outputs of *Stage-1* are a list of global scribbles $\mathcal{S}$ and a *continuous* Text-Energy binary map $\mathcal{B}$ of the complete input image. These are processed by the seam generation pipeline from SeamFormer [31] to obtain tight fitting polygons enclosing the text lines.

We introduce two crucial modifications to the default approach in SeamFormer [31]. Instead of thresholding the binary map, we use output from *Stage-1* as it is, i.e. $\mathcal{B}$ contains floating point values in the range $[0, 1]$. This avoids loss of crucial text presence information caused by thresholding. The second modification is to discard other energy maps used in SeamFormer [31] (smoothing map, diacritic map and sobel map). This eliminates the need for determining energy map weight coefficients. As shown via experiments (Table 3), the quality of our Text-Energy map makes other maps redundant in practice.

4 Implementation Details

Architectural Details: The reference input shape for Stage-1 model is $H_0 \times W_0 \times 3$, where $H_0 = 256$, $W_0 = 256$. The ViT backbone (see Fig. 3) uses a patch size of 16×16, and the output feature map M is of dimension $H \times W \times C$, where $H = 16$, $W = 16$ and $C = 256$, where C is the model's embedding dimension. The encoder consists of 8 encoder layers. Learnable position embeddings are used. The Line-Parameter Generator (Fig. 4) uses $N = 200$ line queries. The transformer decoder consists of $L = 8$ decoder layers. The probability threshold p_0 for line-prediction is set to 0.9. The CNN encoder for Text-Energy Map Generator (Fig. 6) has 4 convolutional layers, each of which reduces the spatial dimensions of the input image by half. The output of this encoder (P) has the same shape as the output of the backbone (M). The transformer decoder consists of $R = 3$ decoder layers. The CNN decoder consists of 4 layers. Each layer consists of a convolutional layer followed by upsampling. In the loss function, we use $\alpha = 0.25$.

Training Details: We perform *Stage-1* training in two phases. In the first phase, we train only the ViT backbone and the Line-Parameter Generator. In the second phase, we freeze both of them, and train only the Text-Energy Map Generator. In the first phase, we use a learning rate of 5×10^{-5}, and train for 60 epochs. We then reduce the learning rate to 10^{-5} and train for another 20 epochs. In the second phase, we train only the energy map generator with a learning rate of 10^{-4} for 50 epochs. We use the AdamW optimizer, with the coefficients set to PyTorch's defaults. Our implementation is based on the distributed PyTorch Lightning framework. We trained our model on 4 NVIDIA RTX 2080Ti GPUs, with 24 images on each GPU. The first phase of training took around 32 hours, and the second phase took around 7 hours.

5 Datasets

To evaluate the proposed model and baselines, we use palm leaf manuscript datasets introduced in earlier works [13,28,31,32] - see Table 1 (shaded blue). In addition, we annotate three new challenging manuscript datasets (WM, UB, SM) for zero-shot evaluation (shaded pink). The diversity in terms of scripts, image aspect ratios, image dimension ranges, number of lines seen in Table 1 underscores the challenge involved in palm-leaf manuscript text line segmentation. The new datasets were annotated using the HInDoLA document image annotation tool [29]. Instead of annotating polygons from scratch, a semi-automatic approach was implemented and integrated into the tool. The annotators added strike-through scribbles for text lines. These scribbles and a binarized version of input image were processed by seam generation module from SeamFormer [31] to obtain text line polygon predictions. The predicted polygon boundaries were adjusted to accommodate missing diacritics and ensure correct enclosure of text lines. Empirically, this semi-automatic approach provided a 75% reduction in annotation time compared to the purely manual variant.

6 Experiments

We compare **LineTR** against various state-of-the-art approaches developed for handwritten historical manuscripts. For fair and consistent comparison, we train all the models (ours, existing approaches) on a single large-scale dataset

Table 1. Dataset Statistics (Sect. 5). Languages: `hin` - Hindi, `tel` - Telugu, `tam` - Telugu, `sun` - Sundanese, `ban` - Balinese, `khm` - Khmer, `mal` - Malayalam, `jav` - Javanese. Newly introduced datasets are shaded pink.

	I2	SD	BL	KH	KG	WM	UB	SM
Name	Indiscapes2	Sundanese	Balinese	Khmer	KgathaM	Wikimedia	Upamiti Bori	SVMP
Train Images	907	31	47	50	313	0	0	0
Test Images	229	30	49	200	79	60	30	30
Avg Lines	11	4	4	5	9	6	5	11
Min Size	442x207	2530x333	2505x637	2244x322	2636x410	2042x1440	2592x1728	4567x1331
Max Size	9184x1064	3159x352	5759x561	8224x696	3404x501	2324x1814	2592x1728	10277x3281
Aspect Ratio	3	9	10	11	7	3.5	1.5	2
Language	`hin,tel,tam`	`sun`	`ban`	`khm`	`mal`	`ban,jav`	`hin`	`tel`
Source	[27]	[28]	[13]	[32]	[31]	Wikimedia	Private	Private

obtained by combining the training sets of existing palm leaf manuscript datasets - Indiscapes2 [27] [I2], KGatham [31] [KG], and Challenge B dataset of ICFHR 2018 Competition On Document Image Analysis Tasks for Southeast Asian Palm Leaf Manuscripts [14] containing manuscripts from Balinese [13] [BL], Khmer [32] [KH], and Sundanese [28] [SD] languages. We report the performance metrics on the respective test sets of these datasets. For existing approaches, we follow the training instructions mentioned in their corresponding papers.

Trivedi et al. [30] demonstrate that Average Hausdorff Distance (AvgHD) is a better measure of prediction performance for line polygon boundaries. Therefore, we report AvgHD in addition to the standard Intersection over Union (IoU) metric. To assess the zero-shot generalizability performance, we report these metrics on three newly introduced datasets unseen during training - WM, UB, and SM - see Table 1 for an overview of the datasets.

7 Results

As Table 2 shows, **LineTR** clearly outperforms other models by a significant margin across all datasets. The consistently poor scores among other baselines is an outcome of loose fit predicted text regions, often missing crucial textual elements such as diacritics. Also, baseline methods [7,20,27] typically approach text line segmentation as a per-pixel classification task, resizing images with large aspect ratios to a fixed lower size. This resizing tends to merge adjacent predicted text, particularly in dense text documents. In other baselines [1,22,31], the suboptimal results are due to excessive dataset-specific decisions. **LineTR**'s numbers on unseen datasets are on par with ones encountered during training. This shows its zero-shot generalization ability.

7.1 Ablations

For ablation experiments, we report metrics on the combined test sets of the seen benchmark datasets - see Table 3. The results suggest that a fine balance

Table 2. Comparative evaluation of **LineTR** using benchmark datasets. All the baseline models are trained on **the pooled dataset** (Sect. 6) using default settings mentioned in respective works to assess their adaptability. The results are reported on test set of each dataset and three additional unseen datasets (zero-shot).

	I2[27]	SD[28]	BL[13]	KH[32]	KG[31]	WM*	UB*	SM*
	AvgHD ↓							
Doc-UFCN [7]	68.60	71.17	74.52	95.90	38.79	41.39	71.10	174.14
SeamFormer [31]	11.82	8.92	104.75	49.03	7.83	11.54	9.46	130.98
Palmira [27]	15.81	6.50	301.21	203.59	7.50	24.11	18.88	15.78
LCG [1]	16.82	39.65	95.18	44.50	29.72	317.98	481.14	1034.88
dhSegment [22]	60.33	66.77	415.24	43.60	319.57	150.43	213.32	414.63
docExtractor [20]	77.25	33.86	43.16	48.36	40.24	87.75	95.51	260.70
LineTR	**1.86**	**1.30**	**22.62**	**14.97**	**2.09**	**0.94**	**1.01**	**3.04**
	IoU ↑							
Doc-UFCN [7]	0.23	0.10	0.08	0.11	0.12	0.15	0.10	0.16
SeamFormer [31]	0.51	0.53	0.32	0.37	0.49	0.49	0.76	0.43
Palmira [27]	0.72	0.66	0.39	0.41	0.62	0.53	0.54	0.69
LCG [1]	0.37	0.12	0.12	0.18	0.20	0.02	0.01	0.07
dhSegment [22]	0.34	0.12	0.03	0.08	0.12	0.10	0.13	0.09
docExtractor [20]	0.03	0.01	0.00	0.02	0.12	0.03	0.03	0.02
LineTR	**0.80**	**0.73**	**0.62**	**0.69**	**0.81**	**0.66**	**0.82**	**0.82**

* Newly introduced datasets with zero-shot baseline testing

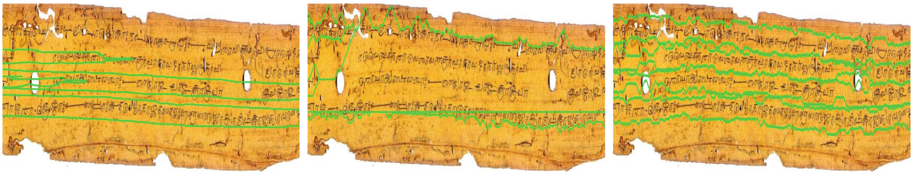
is required so that number of *Line Queries* (N) in 'Line-Parameter Generator' (Sect. 3.2) is neither too many nor too few. Compared to using an arbitrarily threshold binary map similar to SeamFormer [31]'s approach, our unthresholded text energy map (Sect. 3.3) is a noticeably better choice. This is due to the loss of text-information caused due to a fixed threshold value. To demonstrate the importance of context-adaptive patching (Sect. 3.4.1) for generating training data, we considered patches of fixed-size, similar to SeamFormer [31]. However, we found that the combined dataset training is extremely unstable. In fact, the optimization did not even converge. Finally, we replace $\mathcal{L}_{\text{geom}}$ (Sect. 3.2.3) by the EA-loss [36]. This setting also caused the network to not converge.

7.2 Qualitative Results

As Fig. 8 shows, both of **LineTR**'s closest competitors – Palmira [27] and SeamFormer [31] – fail when the text-lines have a curvature spread across the document width. But **LineTR** is able to detect all the text-lines accurately. Similarly,

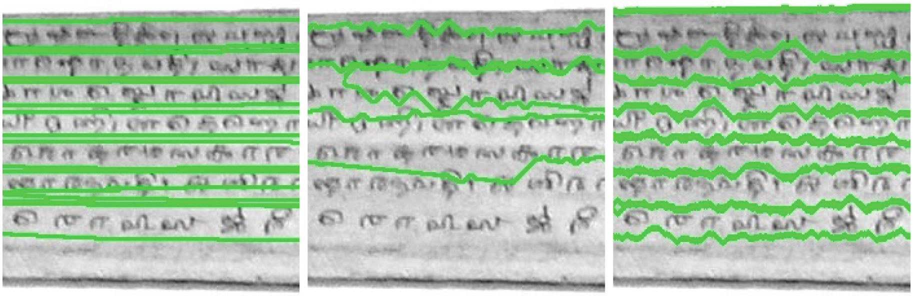
Table 3. Performance scores for **LineTR** ablative variants. (Sect. 7.1). DNC = Did not converge

Ablation Type	Pipeline Component	Ablation Details	IoU ↑	Avg HD ↓
Architectural	Line-Parameter Generator	$N = 100$	0.45	145.39
		$N = 300$	0.61	85.33
		$N = 400$	0.62	28.72
	Text-Energy Map Generator	Threshold the energy map [31]	0.53	8.14
Optimization	Line-Parameter Generator	Replace $\mathcal{L}_{geom}$ by EA-loss [36]	*DNC*	*DNC*
Dataset	Data-Preparation	Sample patches of fixed size	*DNC*	*DNC*
LineTR			**0.74**	**7.13**



(a) Palmira[27] (b) SeamFormer[31] (c) **LineTR**

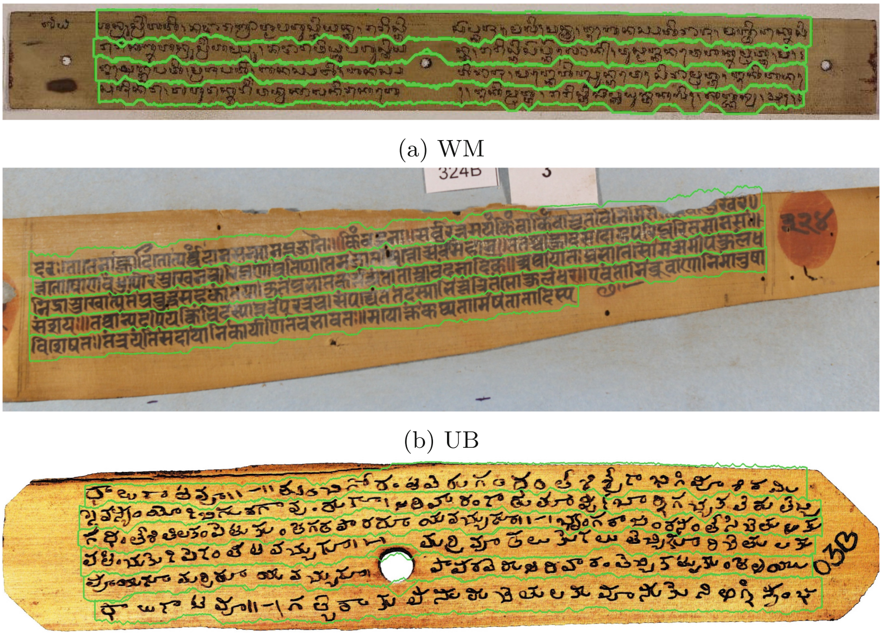
Fig. 8. Performance comparison on a challenging image with curved text-lines



(a) Palmira[27] (b) SeamFormer[31] (c) **LineTR**

Fig. 9. Performance comparison on a challenging image with very dense text

LineTR outperforms Palmira and SeamFormer on images where the density of text is very high (see Fig. 9). Figure 10 shows the zero-shot outputs of **LineTR** on the newly introduced datasets.



(a) WM

(b) UB

(c) SM

Fig. 10. Zero-shot outputs of **LineTR** on the newly introduced datasets

8 Conclusion

LineTR is a novel dataset-agnostic approach for robust text line segmentation in diverse and challenging historical manuscripts. Similar to recent successful approaches, we use a two stage approach - scribble generation and scribble-conditioned polygon generation. However, our unique and novel design choices make a significant difference. The choice of predicting per-patch scribble line parameters in the first stage helps avoid the difficulties of pixel-based scribble representation. Our adaptive patch extraction ensures sufficient context capture for predicting scribble line parameters. A sensible design for Text-Energy Map Generator not only simplifies second stage processing, it also improves overall results.

Unlike existing approaches, **LineTR**'s methodology does not require dataset-specific fine-tuning. Another distinction is that the training process results in a single model and does not require dataset-specific models. These features make **LineTR** advantageous from a maintainability and scalability point of view. **LineTR** not only outperforms strong baselines but also exhibits good zero-shot performance on unseen datasets. This showcases its generalizability and utility for the community.

Acknowledgements. This work is supported by Ministry of Electronics and Information Technology (MeiTY), Government of India.

References

1. Alberti, M., Vögtlin, L., Pondenkandath, V., Seuret, M., Ingold, R., Liwicki, M.: Labeling, cutting, grouping: an efficient text line segmentation method for medieval manuscripts. In: 2019 IPAR (ICDAR), pp. 1200–1206. IEEE (2019)
2. Arvanitopoulos, N., Süsstrunk, S.: Seam carving for text line extraction on color and grayscale historical manuscripts. In: 2014 14th, pp. 726–731. IEEE (2014)
3. Asi, A., Saabni, R., El-Sana, J.: Text line segmentation for gray scale historical document images. In: Proceedings of the 2011 Workshop on Historical Document Imaging and Processing, pp. 120–126 (2011)
4. Avidan, S., Shamir, A.: Seam carving for content-aware image resizing. In: ACM SIGGRAPH 2007 Papers, pp. 10–es (2007)
5. Barakat, B., Droby, A., Kassis, M., El-Sana, J.: Text line segmentation for challenging handwritten document images using fully convolutional network. In: 2018 16th (ICFHR), pp. 374–379. IEEE (2018)
6. Barakat, B.K., et al.: Unsupervised deep learning for text line segmentation. In: 2020 25th (ICPR), pp. 2304–2311. IEEE (2021)
7. Boillet, M., Kermorvant, C., Paquet, T.: Multiple document datasets pre-training improves text line detection with deep neural networks. In: 2020 25th (ICPR), pp. 2134–2141. IEEE (2021)
8. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: ECCV (2020)
9. Chamchong, R., Fung, C.C.: Text line extraction using adaptive partial projection for palm leaf manuscripts from Thailand. In: ICFHR (2012)
10. Dosovitskiy, A., et al.: Image is worth 16×16 words: transformers for image recognition. In: ICLR (2021)
11. Grüning, T., Leifert, G., Strauß, T., Michael, J., Labahn, R.: A two-stage method for text line detection in historical documents. (IJDAR) **22**(3), 285–302 (2019)
12. Jindal, A., Ghosh, R.: Text line segmentation in Indian ancient handwritten documents using faster R-CNN. In: MTA, pp. 1–20 (2022)
13. Kesiman, M.W.A., Burie, J.C., Wibawantara, G.N.M.A., Sunarya, I.M.G., Ogier, J.M.: Amadi_lontarset: the first handwritten balinese palm leaf manuscripts dataset. In: 2016 15th (ICFHR), pp. 168–173. IEEE (2016)
14. Kesiman, M.W.A., et al.: ICFHR 2018 competition on document image analysis tasks for southeast Asian palm leaf manuscripts (2018)
15. Kiessling, B.: Curt: end-to-end text line detection in historical documents with transformers, pp. 34–48. Springer (2022)
16. Kurar Barakat, B., Cohen, R., Droby, A., Rabaev, I., El-Sana, J.: Learning-free text line segmentation for historical handwritten documents. Appl. Sci. (2020)
17. Li, D., Wu, Y., Zhou, Y.: Linecounter: learning handwritten text line segmentation by counting. In: ICIP (2021)
18. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV, pp. 2980–2988 (2017)
19. Mechi, O., Mehri, M., Ingold, R., Amara, N.E.B.: Text line segmentation in historical document images using an adaptive u-net architecture. In: 2019 IPAR (ICDAR), pp. 369–374. IEEE (2019)

20. Monnier, T., Aubry, M.: docExtractor: an off-the-shelf historical document element extraction. In: ICFHR (2020)
21. Nguyen, T.N., Burie, J.C., Le, T.L., Schweyer, A.V.: An effective method for text line segmentation in historical document images. In: ICPR. IEEE (2022)
22. Oliveira, S.A., Seguin, B., Kaplan, F.: dhSegment: a generic deep-learning approach for document segmentation. In: 2018 16th (ICFHR) (2018)
23. Paulus, E., Burie, J.C., Verbeek, F.J.: Text line extraction strategy for palm leaf manuscripts. Pattern Recogn. Lett. **174**, 10–16 (2023)
24. Prusty, A., Aitha, S., Trivedi, A., Sarvadevabhatla, R.K.: Indiscapes: instance segmentation networks for layout parsing of historical Indic manuscripts. In: ICDAR, pp. 999–1006 (2019)
25. Renton, G., Soullard, Y., Chatelain, C., Adam, S., Kermorvant, C., Paquet, T.: Fully convolutional network with dilated convolutions for handwritten text line segmentation. (IJDAR) **21**, 177–186 (2018)
26. Saabni, R., El-Sana, J.: Language-independent text lines extraction using seam carving. In: 2011 IPAR, pp. 563–568. IEEE (2011)
27. Sharan, S., Aitha, S., Kumar, A., Trivedi, A., Augustine, A., Sarvadevabhatla, R.K.: Palmira: a deep deformable network for instance segmentation of dense and uneven layouts in handwritten manuscripts. In: ICDAR (2021)
28. Suryani, M., Paulus, E., Hadi, S., Darsa, U.A., Burie, J.C.: The handwritten sundanese palm leaf manuscript dataset from 15th century. In: 2017 14th IAPR IPAR (ICDAR), vol. 1, pp. 796–800. IEEE (2017)
29. Trivedi, A., Sarvadevabhatla, R.K.: Hindola: a unified cloud-based platform for annotation, visualization and machine learning-based layout analysis of historical manuscripts. In: ICDARW, vol. 2, pp. 31–35. IEEE (2019)
30. Trivedi, A., Sarvadevabhatla, R.K.: Boundarynet: an attentive deep network for semi-automatic layout annotation. In: ICDAR (2021)
31. Vadlamudi, N., Krishna, R., Sarvadevabhatla, R.K.: Seamformer: high precision text line segmentation for handwritten documents. In: IPAR, pp. 313–331. Springer (2023)
32. Valy, D., Verleysen, M., Chhun, S., Burie, J.C.: A new Khmer palm leaf manuscript dataset for document analysis and recognition: sleukrith set. In: International Workshop on Historical Document Imaging and Processing (2017)
33. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
34. Wang, Y., Zhang, X., Yang, T., Sun, J.: Anchor DETR: query design for transformer-based detector. In: AAAI, vol. 36, pp. 2567–2575 (2022)
35. Wigington, C., Tensmeyer, C., Davis, B., Barrett, W., Price, B., Cohen, S.: Start, follow, read: End-to-end full-page handwriting recognition. In: ECCV (2018)
36. Zhao, K., Han, Q., Zhang, C.B., Xu, J., Cheng, M.M.: Deep Hough transform for semantic line detection. IEEE TPAMI **44**(9), 4793–4806 (2021)
37. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)

Region-Level Layout Generation for Multi-level Pre-trained Model Based Visual Information Extraction

Shuai Li[1], Xiao-Hui Li[2], Fei Yin[2], and Lin-Lin Huang[1](✉)

[1] BeiJing JiaoTong University, Beijing 100044, China
huangll@bjtu.edu.cn

[2] State Key Laboratory of Multimodal Artificial Intelligence Systems,
Institute of Automation of Chinese Academy of Sciences, Beijing 100190, China

Abstract. Multimodal pre-trained models have made significant advancements in the field of visual information extraction by jointly modeling textual, layout, and visual modalities, among which the layout information plays a key role in modeling document inherent structures. However, due to the diversity and complexity of document types and typography styles, it is still not fully studied on how to better model various document layouts comprehensively and hierarchically. Compared with single-level layout adopted by most previous works, multi-level layouts including word-level, segment-level, and region-level layouts can provide a more scientifically modeling of complex document structures. Considering that most existing OCR tools lack region-level layout outputs of high quality, which poses challenges for the utilization of multi-level layout information, we thus propose a region-level layout generation method named **ReMe** based on hierarchical clustering. By iteratively clustering and merging segment-level bounding boxes, **ReMe** aims to ensure that semantically related segments with strong correlations share the same region-level bounding boxes. ReMe can be seamlessly integrated into the existing multi-level layout information modeling methods with negligible cost. Experimental results show that after pretrained with only 2 million documents from the IIT-CDIP dataset, the model can achieve new state of the art results on downstream visual information extraction datasets, and the region-level layout information generated by **ReMe** can significantly enhance the model's understanding of structured documents, especially the performance on the Relation Extraction task.

Keywords: visual information extraction · multimodal · region level · text-layout

1 Introduction

Visual Information Extraction(VIE) is a critical document understanding topic that aims at reading and analyzing the scanned or digital-born documents, and extracts text information of specified categories. In recent years, with the

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 234–249, 2025.
https://doi.org/10.1007/978-3-031-78495-8_15

advancement of deep learning techniques, particularly the Transformer architecture [22], VIE has entered a new stage of development [4,5,11,16,18,23]. Most of these works can be traced back to BERT [9], which introduces a "Masked Language Modeling(MLM)" task based on the Transformer encoder [22] structure. It effectively explores deep semantic features, achieving outstanding results in the field of Natural Language Processing(NLP). Unlike natural language, the diversity in document layouts and formats has always been a significant challenge in the VIE. How to better model the multimodal information involved in documents has thus become a key research direction.

Based on BERT [9], many models including LayoutLMs [7,26,27], DocFormer [1], and others [14,20,30] incorporate textual with visual features and layout information. However, the incorporation of visual information significantly increases the training cost, implying that the model requires more training data and larger memory. Nonetheless, when dealing with text-centric documents, such as forms, contracts, receipts, and invoices, visual features are not always essential. Consequently, some researchers contemplate modeling such documents solely using textual and layout information.

The layout information in text-centric structured documents can be divided into word-level, segment-level, and region-level from fine-grained to coarse-grained [25]. Models such as StructuralLM [13], Bros [6], and others [21,24] model text-centered structured documents using textual and layout information, avoiding a dramatic increase in computational complexity caused by the introduction of visual features. These models have demonstrated competitive performance in downstream tasks. Actually, these models only utilize single-level layout information in practice.

In the absence of visual features, multi-level layout information should be utilized even more fully. As described in ERNIE-mmLayout [25], the lack of natural region-level layout information in raw data or OCR tool results poses challenges for the utilization of multi-level layout information. Although the recent GraphMLLM [3] has achieved promising results by modeling text and multi-level layout information, it oversimplifies the process by merging all segment-level layout information in the document into a single region-level layout bounding box. Such an approach is overly simplistic, and the resulting region provides limited assistance for intelligent VIE.

In this paper, we propose **ReMe**, a method to generate high quality region layout information based on a modified hierarchical clustering algorithm. Initially, segment-level layout provided by the OCR engine information is clustered using a low distance threshold to generate primary region-level bounding boxes. Subsequently, these primary regions are merged according to specific heuristic rules to produce the final region-level layout information. This approach ensures that semantically related entities are grouped within the same region as much as possible, thus providing support for modeling complex document structures using text and multi-level layout information. To validate the effectiveness of the region-level layout information, we combine **ReMe** with GraphMLLM [3] and perform Semantic Entity Recognition(SER) and Relationship Extraction(RE) tasks on the FUNSD [8], XFUND [29], and CORD [19] datasets. Experiments show that the incorporation of region-level layout information significantly improves the performance of GraphMLLM on the FUNSD, XFUND, and CORD datasets, especially in the RE task. Specifically, under Language-specific fine-tuning conditions, the average F1 score improved by 3.87. Under Multi-task fine-tuning conditions, the average F1 score improved by 1.67. And under Cross-lingual zero-shot conditions, the average F1 score improved by 3.41.

The contributions of this paper are summarized as follows:

- We propose a region-level layout information generation method ReMe based on the modified hierarchical clustering algorithm, which effectively alleviates the problem of lacking region-level layout information in manual annotations or OCR results, and provides support for subsequent multi-level layout information modeling.
- ReMe significantly improves the ability of the multi-level pre-trained model to understand document layout information, thereby enhancing the model's cross-language understanding capability. Using only layout and textual information, it achieves new state-of-the-art results in SER and RE results on FUNSD, XFUND and CORD datasets.

2 Related Work

Since documents usually have complex hierarchical layout structures, existing pre-trained models can be roughly classified into two types based on the layout information they adopt: single-level layout based model and multi-level layout based model.

2.1 Single-Level Layout Based Model

Based on LayoutLM [27], StructuralLM [13] replaces word-level spatial layout information with segment-level, meaning that words belonging to the same semantic entity share the same layout information. This makes StructuralLM aware of which words are from the same semantic entity, and thus enables the model to capture not only the semantic representation of individual entities but also the spatial relationship between entities. Bros [6] modifies LayoutLM by replacing absolute position encoding with relative position encoding, effectively incorporating the relative positional relationships between word-level bounding boxes into the self-attention mechanism. This enhancement improves the model's capability to capture and utilize text position information. LiLT [24] proposes a two-stream architecture where text and segment-level layout information are processed separately through the text flow and layout flow, respectively. This allows the model to learn layout knowledge from single-language documents and generalize it to multi-language document downstream tasks. LayoutMask [21] adopts local 1D position and segment-level bounding boxes as layout input and can enhance text-layout interactions and layout representation learning during pre-training.

Although these models have achieved outstanding performance using only text and single-level layout information, the relationship between different levels of layout information still needs further exploration.

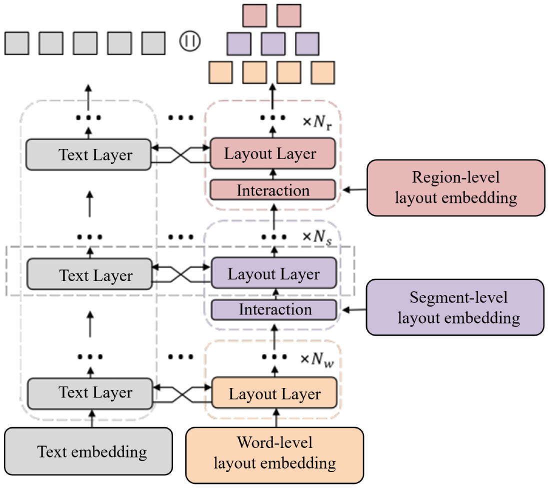


Fig. 1. The overall architecture of GraphMLLM [3].

2.2 Multi-level Layout Based Model

StrucTexT [15] incorporates text and word-level layout bounding boxes as the initial language token embeddings, and combines visual features with segment-level layout bounding boxes to form the initial visual token embeddings. By employing this approach, the integration and exploitation of multi-level layout information are successfully attained. ERINE-mmLayout [25] uses a clustering-based method to detect region-level layout information, and then incorporates the segment-level and region-level information into a LayoutLMv2-based model. This enables the model to capture multi-level layout information, effectively boosting its performance in both Information Extraction and Document Question Answering tasks.

GraphMLLM [3] is a recently introduced model that is graph-based, incorporates multi-level layout information, and is language-independent. Unlike StrucTexT [15] and ERNIE-mmLayout [25], it solely relies on textual and layout modality information for document modeling. As depicted in Fig. 1, GraphMLLM employs a dual-stream structure to separately encode text and layout features. Information is then interacted between these two modalities via an attention-based hierarchical interaction mechanism. The word-level and segment-level layout information is derived from ground truth annotations or

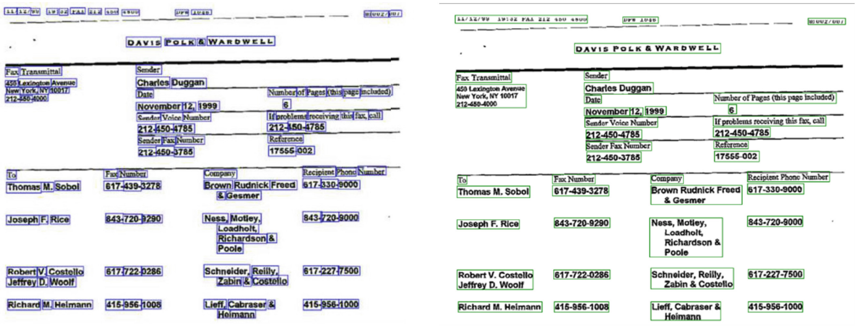
OCR results, while the region-level layout information originates from a bounding box covering all segment-level bounding boxes. However, representing region-level layout information of a document through a single bounding box is overly simplistic. Experimental results also indicate minimal improvement from incorporating this region-level layout information.

3 Methodology

ReMe is a hierarchical clustering method. Firstly, we cluster the segment-level bounding boxes with a low threshold to generate preliminary regions. Then, according to specific rules, these preliminary regions are merged to produce the final region-level bounding boxes.

3.1 Preliminary Clustering

The layout information at the word-level and segment-level is represented in the form of 2D bounding boxes with coordinates (x_0, y_0, x_1, y_1), where (x_0, y_0) corresponds to the position of the upper left corner of the bounding box, and (x_1, y_1) represents the position of the lower right corner.



11/12/99 19:32 FAX 212 450 4500 DPW 1046 002/007

DAVIS POLK & WARDWELL

Fax Transmittal
450 Lexington Avenue
New York, NY 10017
212-450-4000

Sender
Charles Duggan

Date
November 12, 1999

Number of Pages (this page included)
6

Sender Voice Number
212-450-4785

If problems receiving this fax, call
212-450-4785

Sender Fax Number
212-450-3785

Reference
17555-002

To	Fax Number	Company	Recipient Phone Number
Thomas M. Sobol	617-439-3278	Brown Rudnick Freed & Gesmer	617-330-9000
Joseph F. Rice	843-720-9290	Ness, Motley, Loadholt, Richardson & Poole	843-720-9000
Robert V. Costello Jeffrey D. Woolf	617-722-0286	Schneider, Reilly, Zabin & Costello	617-227-7500
Richard M. Heimann	415-956-1008	Lieff, Cabraser & Heimann	415-956-1000

(a) word-level layout

(b) segment-level layout

Fig. 2. (a) word-level and (b) segment-level layout bounding boxes. The document is sourced from the FUNSD [8] dataset.

As illustrated in Fig. 2, segment-level layout information enables models to be aware of which words belong to the same semantic entity. Consider the example of the words "Brown", "Rudnick", "Freed", "&", and "Gesmer" that constitute a company's name. Without segment-level information, it would be challenging to recognize that these individual words collectively form part of the company's name. With the segment-level layout information, models can more easily comprehend this association.

It is clear that there exists a certain correlation or similarity among various semantic entities. Analogous to the relationship between word-level and segment-level information, allowing semantically correlated or similar entities to share the same region-level layout information will enable the model to learn richer semantic information. Inspired by mmLayout [25], we apply a density-based clustering method DBSCAN [10] to generate region-level layout information. Given two segment-level bounding boxes $(x_0^0, y_0^0, x_0^1, y_0^1)$ and $(x_1^0, y_1^0, x_1^1, y_1^1)$, the distance of their bounding boxes can be defined as follows:

$$dist(i,j) = \sqrt{(dist_x(i,j))^2 + (dist_y(i,j))^2}, \tag{1}$$

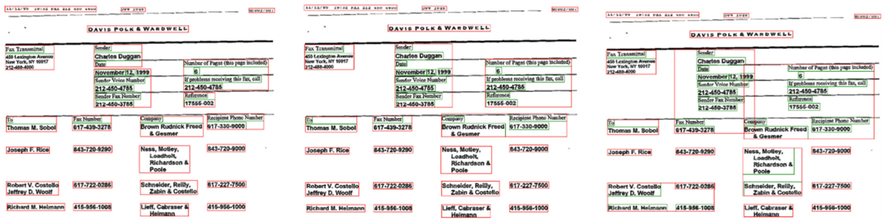
where

$$dist_x(i,j) = \max(\max(x_i^0, x_j^0) - \min(x_i^1, x_j^1), 0), \tag{2}$$

and

$$dist_y(i,j) = \max(\max(y_i^0, y_j^0) - \min(y_i^1, y_j^1), 0). \tag{3}$$

$dist_x$ and $dist_y$ represent the horizontal and vertical distance between the two boxes. When $dist < r$, where r is a hyperparameter, two segment-level bounding boxes can be clustered into a region-level bounding box.



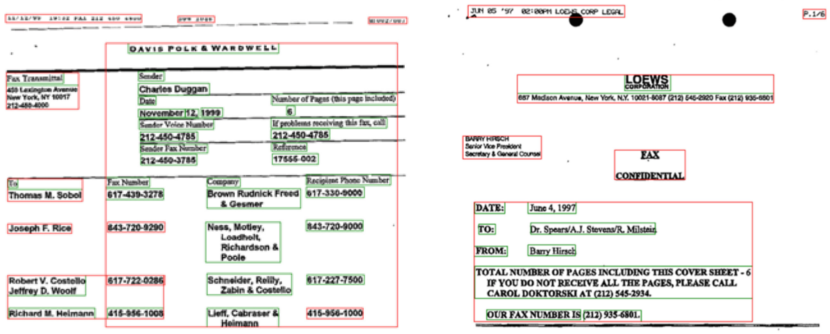
(a) $r = 5$ (b) $r = 10$ (c) $r = 20$

Fig. 3. Region-level layout bounding boxes (red) generated by preliminary clustering with different values of r in FUNSD. The green means the original segment-level layout bounding boxes. (Color figure online)

As demonstrated in Fig. 3, when r is too small, the resulting regions tend to be excessively fragmented, causing many semantic entities that ought to belong to the same region to be divided into separate regions. Conversely, a large value of r leads to oversized regions, where many semantically weakly associated entities are grouped together within the same region. More importantly, the existence of overlapping and covering among regions can easily cause semantic confusion.

For a particular document image, there may exist an appropriate hyperparameter r that yields the most reasonable regions. However, this value of r may not be suitable for all document images. As demonstrated in Fig. 4, when $r =$

30, the resulting regions appear more reasonable for the document on the right, which is consistent with the illustrations presented in mmLayout [25]. However, for the document on the left, it may not be an optimal choice. When conducting pre-training on large number of documents, it is impractical to find an appropriate hyperparameter r for all individual document images.



DAVIS POLK & WARDWELL

Fax Transmittal
450 Lexington Avenue
New York, NY 10017
212-450-4000

Sender
Charles Duggan

Date
November 12, 1999

Number of Pages (this page included)
6

Sender Voice Number
212-450-4785

If problems receiving this fax, call
212-450-4785

Sender Fax Number
212-450-3785

Reference
17555-002

To	Fax Number	Company	Recipient Phone Number
Thomas M. Sobol	617-439-3278	Brown Rudnick Freed & Gesmer	617-330-9000
Joseph F. Rice	843-720-9290	Ness, Motley, Loadholt, Richardson & Poole	843-720-9000
Robert V. Costello Jeffrey D. Woolf	617-722-0286	Schneider, Reilly, Zabin & Costello	617-227-7500
Richard M. Heimann	415-956-1008	Lieff, Cabraser & Heimann	415-956-1000

(a)

JUN 05 '97 02:00PM LOEWS CORP LEGAL P.1/6

LOEWS
CORPORATION
667 Madison Avenue, New York, N.Y. 10021-8087 (212) 545-2920 Fax (212) 935-6801

BARRY HIRSCH
Senior Vice President
Secretary & General Counsel

FAX
CONFIDENTIAL

DATE: June 4, 1997

TO: Dr. Spears/A.J. Stevens/R. Milstein

FROM: Barry Hirsch

TOTAL NUMBER OF PAGES INCLUDING THIS COVER SHEET - 6
IF YOU DO NOT RECEIVE ALL THE PAGES, PLEASE CALL
CAROL DOKTORSKI AT (212) 545-2934.

OUR FAX NUMBER IS (212) 935-6801.

(b)

Fig. 4. Region-level layout bounding boxes (red) of different documents generated by preliminary clustering with $r = 30$ in FUNSD. The green means the original segment-level layout bounding boxes. (Color figure online)

3.2 Regions Merging

As mentioned above, single-level clustering alone cannot provide high-quality region-level layout information. When clustering is performed using a low threshold r, spatially adjacent entities are grouped into the same region, resulting in minimal overlap or crossover among regions. However, the resulting regions are often too fragmented, and such fragmented regions provide limited assistance in modeling high-level layout information.

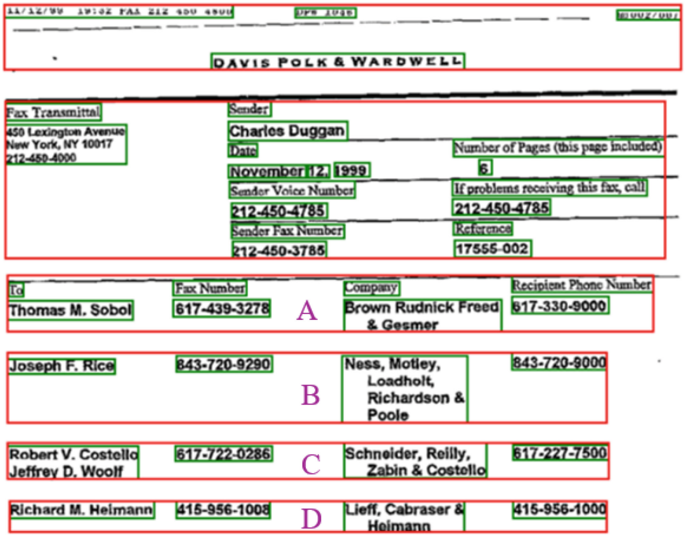
To obtain higher-quality region-level information, small regions can be merged according to certain rules. To avoid semantic confusion during the merging process, it is essential to ensure that there are no overlaps or intersections among the newly generated region-level information. For highly structured documents such as forms, invoices, and contracts, the correlation between two entities is generally proportional to their spatial distance, and the reading order typically follows a left-to-right and top-to-bottom sequence. When merging small regions generated by low-threshold clustering, the merging process can also be performed based on spatial distance, following the left-to-right and top-to-bottom order.

After preliminary clustering, a sequence of bounding boxes representing small regions can be obtained. Ordered from left-to-right and top-to-bottom according to the coordinates of the top left corner, they are denoted as $\{b_1, \dots, b_n\}$. The merging process can be described as follows:

(1) ReMe starts with b_1. According to Eq. (1), we calculate the spatial distances between b_1 and the remaining bounding boxes. These distances are then sorted in ascending order, with the corresponding bounding boxes denoted as $\{d_2, \dots, d_n\}$;
(2) Merge b_1 and d_2 to form a new bounding box, denoted as R_1;
(3) Sequentially calculate the spatial positional relationship between R_1 and $\{d_3, \dots, d_n\}$. If there is no intersection or overlap between R_1 and $\{d_3, \dots, d_n\}$, the merging is considered successful. If there is any intersection or overlap, the merging is deemed unsuccessful, and in this unsuccessful case, the value of b_1 is assigned to R_1;
(4) Update the bounding boxes sequence to $\{R_1, d_3, \dots, d_n\}$;
(5) Iteratively repeat steps (2), (3), and (4), progressively merging bounding boxes until the sequence is updated to only $\{R_1\}$, which represents the final region-level bounding box generated by ReMe;
(6) Update the original sequence $\{b_1, \dots, b_n\}$ by replacing b_1 with R_1, resulting in the new sequence $\{R_1, b_2, \dots, b_n\}$. Subsequently, employ Eq. (1) to ascertain whether each b_i (i = 2, 3, ..., n) is contained within R_1. In the event that any b_i is determined to be enclosed by R_1, it shall be eliminated from the sequence;

Repeat steps (1) to (6) until all b_i (i = 2, 3, ..., n) are removed. At this point, the sequence $\{R_1, R_2, \dots, R_m\}$ represents the region-level bounding boxes. The pseudo-code is shown in Algorithm 1. In the preliminary clustering phase, we set the hyperparameter r to 10. After applying ReMe to Fig. 2(b), the resulting region-level bounding boxes are illustrated in Fig. 5. The semantic entities in Boxes A, B, C, and D all follow the logic of "To", "Fax Number", "Company", and "Recipient Phone Number". Compared to Fig. 3 and Fig. 4, region-level bounding boxes generated by ReMe are more reasonable.

ReMe can be regarded as an improvement of the single-level clustering algorithm and becomes a hierarchical clustering method. In single-level clustering algorithms, the spatial distance is used as a constraint to determine whether two segment-level bounding boxes can be merged into one. Contrastively, algorithms based on hierarchical merging rules only rely on spatial distance to determine the merging order, and whether two bounding boxes can be successfully merged does not directly depend on the distance between them. Only when the new region generated after merging does not overlap or intersect with other bounding boxes, the merging of the two boxes is allowed.



11/12/99 19:02 FAX 212 450 4500 DPW 1048 002/007

DAVIS POLK & WARDWELL

Fax Transmittal

450 Lexington Avenue
New York, NY 10017
212-450-4000

Sender
Charles Duggan

Date
November 12, 1999

Number of Pages (this page included)
6

Sender Voice Number
212-450-4785

If problems receiving this fax, call
212-450-4785

Sender Fax Number
212-450-3785

Reference
17555-002

To	Fax Number	Company	Recipient Phone Number
Thomas M. Sobol	617-439-3278	Brown Rudnick Freed & Gesmer	617-330-9000
Joseph F. Rice	843-720-9290	Ness, Motley, Loadholt, Richardson & Poole	843-720-9000
Robert V. Costello Jeffrey D. Woolf	617-722-0286	Schneider, Reilly, Zabin & Costello	617-227-7500
Richard M. Heimann	415-956-1008	Lieff, Cabraser & Heimann	415-956-1000

Fig. 5. region-level bounding boxes(red) generated by ReMe. (Color figure online)

Algorithm 1. ReMe Algorithm

```
Require: A sequence of segment-level bounding boxes $\{s_1, \ldots, s_k\}$, low threshold $r$
Ensure: Region-level bounding boxes $\{R_1, \ldots, R_m\}$
 1: Cluster $\{s_1, \ldots, s_k\}$ to obtain B $= \{b_1, \ldots, b_n\}$ {low threshold $r$}
 2: for $i = 1$ to $n$ do
 3:     Calculate distances between $b_1$ and $\{b_2, \ldots, b_n\}$ to obtain
        $\{d_2, \ldots, d_n\}$ {Equation(1)}
 4:     Merge $b_1$ and $d_2$ to form $R_1$
 5:     for $j = 3$ to $n$ do
 6:         if No intersection or overlap between $R_1$ and $d_j$ then
 7:             Continue
 8:         else
 9:             $R_1 \leftarrow b_1$
10:             break
11:         end if
12:     end for
13:     Update bounding boxes sequence to $\{R_1, d_3, \ldots, d_n\}$
14: end for
15: Continue merging until the sequence is updated to only $\{R_1\}$
16: for $i = 2$ to $n$ do
17:     if $b_i$ is enclosed by $R_1$ then
18:         Remove $b_i$ from sequence B
19:     end if
20: end for
21: Repeat steps2 to steps20 until all $b_i$ are removed from the sequence.
22: return $\{R_1, R_2, \ldots, R_m\}$ as region-level bounding boxes
```

4 Experiments

4.1 Datasets and Evaluation

The experiments in this work involve two tasks, namely Semantic Entity Recognition(SER) and Relationship Extraction(RE). All experiments are conducted on the FUNSD [8], XFUND [29] and CORD [19] datasets. The SER aims to assign each semantic entity a label among "question", "answer", "header" or "other" and RE is the task of extracting the relationship between entities.

The performance of the pre-trained model is measured by entity-level F1 in three main settings: (1)Language-specific fine-tuning, which means fine-tuning and testing on a specific language; (2)Zero-shot transfer learning, which means fine-tuning on English data only and testing on multilingual dataset; (3)Multi-task fine-tuning, which means fine-tuning on multilingual dataset and testing on individual language data.

4.2 Settings

We adopt the recently proposed GraphMLLM [3] as the backbone to integrate with ReMe. To more fairly demonstrate the effectiveness of ReMe, we also use the same pre-training tasks and hyperparameters described in GraphMLLM. We set the batch size of 48 and train the model for 2 epochs on 2M documents randomly selected from the IIT-CDIP dataset [12] using 2 NVIDIA A800 80GB GPUs. Adam optimizer with the learning rate $2e^{-5}$, weight decay $1e^{-2}$, and (beta1, bata2) = (0.9, 0.999) are also used in pre-training.

4.3 Main Results

Language-Specific Fine-Tuning. We first evaluate ReMe on FUNSD and CORD, and the results are shown in Table 1. It is concluded that ReMe is effective as an independent region-level layout generator. By utilizing high-quality region-level layout information generated by ReMe, GraphMLLM [3] surpasses its original performance in the SER task across two datasets.

Similar to LiLT [24], GraphMLLM initializes the text flow from the existing pre-trained English RoBERTa$_{base}$ [17] for pre-training. To fine-tune on non-English document data, RoBERTa$_{base}$ should be replaced with InfoXLM$_{base}$ [2].

Then we evaluate ReMe on language-specific fine-tuning tasks(fine-tuning on X, testing on X) of FUNSD and XFUND. The results are shown in Table 2, by utilizing ReMe, GraphMLLM [3] has achieved the highest F1 scores on both the SER and RE tasks of each language. This significant improvement demonstrates that high-quality region-level layout information can aid models in acquiring a greater amount of language-independent knowledge, which can then be transferred from pre-training to downstream tasks.

Table 1. English dataset fine-tuning results. "Doc" represents the total number of documents images utilized in pre-training, measured in millions(M). "GraphMLLM+R" indicates that the model employs region-level layout information generated by ReMe. **Bold** indicates the SOTA and underline indicates the second best. "+/−" represents the improvement of GraphMLLM [3] performance by combining with Reme.

model	Docs	FUNSD			CORD		
		Precision	Recall	F1	Precision	Recall	F1
RERT$_{base}$ [9]	–	54.96	67.1	60.26	88.33	91.07	89.68
RoBERTa$_{base}$ [17]	–	63.49	69.75	66.48	–	–	–
LayoutLM$_{base}$ [27]	11M	75.97	81.55	78.66	94.37	95.08	94.72
LayoutLMv3$_{base}$ [7]	11M	–	–	**90.29**	–	–	96.56
GraphDoc$_{base}$ [30]	0.32M	–	–	87.95	–	–	**96.93**
LayoutXLM$_{base}$ [28]	30M	–	–	79.40	–	–	–
LiLT$_{base}$ [24]	11M	87.21	**89.65**	88.41	95.98	96.16	96.07
GraphMLLM [3]	2M	86.16	88.40	87.27	95.37	95.58	95.48
GraphMLLM+R	2M	**88.00**	88.70	88.35	**96.19**	**96.33**	96.26
+/−	–	1.84	0.3	1.08	0.82	0.75	0.78

Table 2. language-specific fine-tuning F1 accuracy on FUNSD and XFUND.

task	Model	Pretrain Docs		FUNSD	XFUND							Avg.
		Language	Size	EN	ZH	JA	ES	FR	IT	DE	PT	
SER	LayoutXLM [28]	Multilingual	30M	79.40	89.24	79.21	75.50	79.02	80.80	82.22	79.03	80.56
	LiLT [24]	English only	11M	84.15	89.38	79.64	79.11	79.53	83.76	82.31	82.20	82.51
	GraphMLLM [3]	English only	2M	84.03	90.80	80.34	79.54	83.74	84.58	84.81	83.01	83.86
	GraphMLLM+R	English only	2M	**85.38**	**91.36**	**80.40**	**81.79**	**84.36**	**86.43**	**86.30**	**84.15**	**85.15**
	+/−	–	–	1.35	0.56	0.06	2.25	0.65	1.85	1.49	1.14	1.17
RE	LayoutXLM [28]	Multilingual	30M	54.83	70.73	69.63	68.96	63.53	64.15	65.51	57.18	64.32
	LiLT [24]	English only	11M	62.76	72.97	70.37	71.95	69.65	70.43	65.58	58.74	67.81
	GraphMLLM [3]	English only	2M	64.62	77.34	71.78	68.32	67.81	71.72	67.44	58.88	68.49
	GraphMLLM+R	English only	2M	**67.33**	**81.46**	**75.37**	**73.52**	**74.04**	**74.20**	**70.17**	**62.30**	**72.30**
	+/−	–	–	2.71	4.12	3.59	5.20	6.23	2.48	2.73	3.42	3.81

Zero-Shot Transfer Learning. Table 3 presents the results of cross-language zero-shot transfer learning(fine-tuning on FUNSD, testing on X). Neither during the pre-training nor the fine-tuning phases did GraphMLLM [3] encounter non-English documents. Clearly, by combining with ReMe, the GraphMLLM model transfers the most language-independent knowledge from English to other languages. However, when evaluating documents in Spanish (ES) and Portuguese (PT), ReMe led to a performance degradation on the SER task. This observation suggests that language-independent knowledge gained through a specialized task on a particular corpus might not universally be applicable to all datasets. Nonetheless, considering the average results, ReMe significantly improved the model's performance. Hence, despite isolated instances of performance deterioration, the utility of ReMe cannot be discounted.

Table 3. Cross-lingual zero-shot transfer F1 accuracy on FUNSD and XFUND.

task	Model	Pretrain Docs		FUNSD	XFUND							Avg.
		Language	Size	EN	ZH	JA	ES	FR	IT	DE	PT	
SER	LayoutXLM [28]	Multilingual	30M	79.40	60.19	47.15	45.65	57.57	48.46	52.52	53.90	55.61
	LiLT [24]	English only	11M	84.15	61.52	51.84	51.01	59.23	53.71	**60.13**	63.25	60.61
	GraphMLLM [3]	English only	2M	84.03	61.02	51.18	**51.04**	60.30	54.46	58.54	**63.87**	60.56
	GraphMLLM+R	English only	2M	**85.38**	**66.02**	**52.23**	50.26	**61.90**	**58.87**	59.26	61.56	**61.94**
	+/−	–	–	1.35	5.00	1.05	-0.78	1.60	4.41	0.72	-2.31	1.38
RE	LayoutXLM [28]	Multilingual	30M	54.83	44.94	44.08	47.08	44.16	40.90	38.20	36.85	43.88
	LiLT [24]	English only	11M	62.76	47.64	50.81	49.68	52.09	46.97	41.69	42.72	49.30
	GraphMLLM [3]	English only	2M	64.62	56.67	58.11	54.53	58.52	50.22	49.12	43.30	54.39
	GraphMLLM+R	English only	2M	**67.33**	**61.23**	**58.62**	**57.69**	**61.92**	**55.76**	**50.35**	**49.48**	**57.80**
	+/−	–	–	2.71	4.56	0.51	3.16	3.40	5.54	1.23	6.18	3.41

Multi-task Fine-Tuning. Table 4 shows the results of multi-task fine-tuning(fine-tuning on 8 languages all, testing on X). Specifically, it demonstrates the performance of GraphMLLM that was pre-trained solely on an English dataset and subsequently fine-tuned concurrently on datasets derived from eight distinct languages. The model was then evaluated using datasets from each corresponding language. The results reveal that when the model handles documents in multiple languages simultaneously, it can also reap the benefits of incorporating ReMe.

4.4 Ablation Studies

For better understanding the performance of ReMe, we conduct ablation studies to determine how region-level layout information of varying quality affects the performance of GraphMLLM. The experiments were conducted on FUNSD and XFUND datasets, using the language-specific task setting.

As demonstrated in Table 5 and Table 6, the region-level layout information generated by ReMe proves to be more efficacious in enhancing model performance. That is due to the fact that the granularity of regions generated by preliminary clustering is too small to capture the high-level layout information of

Table 4. Multitask fine-tuning F1 accuracy on FUNSD and XFUND.

task	Model	Pretrain Docs		FUNSD	XFUND							Avg.
		Language	Size	EN	ZH	JA	ES	FR	IT	DE	PT	
SER	LayoutXLM [28]	Multilingual	30M	79.24	87.55	79.64	77.98	81.73	82.10	83.22	82.41	82.01
	LiLT [24]	English only	11M	85.74	90.47	80.88	83.40	85.77	87.92	87.69	84.93	85.85
	GraphMLLM [3]	English only	2M	87.37	91.13	80.79	85.23	**88.54**	88.06	**88.81**	86.03	87.00
	GraphMLLM+R	English only	2M	**87.94**	**92.49**	**81.40**	**86.53**	86.92	**89.15**	88.80	**86.45**	**87.46**
	+/−	–	–	0.57	1.36	0.61	1.30	-1.62	1.09	-0.01	0.42	0.46
RE	LayoutXLM [28]	Multilingual	30M	66.71	82.41	81.42	81.04	82.21	83.10	78.54	70.44	78.23
	LiLT [24]	English only	11M	74.07	82.41	83.45	83.35	84.66	84.58	78.78	76.43	81.25
	GraphMLLM [3]	English only	2M	82.98	89.46	84.56	85.33	88.60	86.41	83.15	78.36	84.86
	GraphMLLM+R	English only	2M	**87.73**	**90.41**	**85.29**	**86.04**	**90.33**	**88.52**	**83.38**	**80.50**	**86.53**
	+/−	–	–	2.71	4.56	0.51	3.16	3.40	5.54	1.23	6.18	1.67

documents. The original GraphMLLM utilizes bounding boxes encompassing all text to furnish region-level layout information for the entirety of the document. This larger granularity impairs the effective conveyance of structural information pertaining to the document.

Table 5. Comparison on the SER task of FUNSD dataset. "original" means the original region-level layout information(a box covering all segment-level bounding boxes [3]); "clustering" means the region-level layout information generated by Preliminary Clustering (Sect. 3.1); "ReMe" means the region-level layout information generated by ours ReMe.

Task	Regions	Precision	Recall	F1
SER	original	86.16	88.40	87.27
	clustering	87.08	88.60	87.83
	ReMe	88.00	88.70	88.35

Table 6. Comparison F1 accuracy on FUNSD and XFUND.

Task	Regions	EN	ZH	JA	ES	FR	IT	DE	PT	Avg.
RE	original	64.62	77.34	71.78	68.32	67.81	71.72	67.44	58.88	68.49
	clustering	65.26	73.20	68.47	70.62	72.77	71.15	63.75	59.87	68.14
	ReMe	67.33	81.46	75.37	73.52	74.04	74.20	70.17	62.30	72.30

5 Conclusion

The lack of natural region-level layout information in manual annotations or results of OCR tools poses challenges for the utilization of multi-level layout information. To address this issue, we propose **ReMe**, a region-level layout generator based on hierarchical clustering. It aims to ensure that semantically related segments with strong correlations share the same region-level bounding boxes. We conduct extensive experiments on the FUNSD, XFUND, and CORD datasets, with a focus on Semantic Entity Recognition and Relation Extraction tasks, to verity the effectiveness of ReMe. Experimental results under three settings (language-specific, cross-lingual zero-shot transfer, and multi-task fine-tuning) clearly demonstrate that ReMe helps the model to more fully explore multi-level layout information within documents, thereby achieving better performance on various visual information extraction tasks. In the future, we will explore how to introduce more semantic information into the generation of region-level layout information, rather than only the segment-level layout information used in this paper.

Acknowledgements. This work is supported by the National Key Research and Development Program Grant 2020AAA0109700, the Youth Program of State Key Laboratory of Multimodal Artificial Intelligence Systems, and the National Natural Science Foundation of China(NSFC) Grant U23B2029.

References

1. Appalaraju, S., Jasani, B., Kota, B.U., Xie, Y., Manmatha, R.: DocFormer: end-to-end transformer for document understanding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 993–1003 (2021)
2. Chi, Z., et al.: InfoXLM: an information-theoretic framework for cross-lingual language model pre-training. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 3576–3588 (2021)
3. Dai, H., Li, X., Yin, F., Yang, X., Mei, S., Liu, C.: GraphMLLM: a graph-based multi-level layout language-independent model for document understanding. In: International Conference on Document Analysis and Recognition (ICDAR) (2024)
4. Dhouib, M., Bettaieb, G., Shabou, A.: DocParser: end-to-end OCR-free information extraction from visually rich documents. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) ICDAR 2023. LNCS, vol. 14191, pp. 155–172. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41734-4_10
5. He, J., et al.: ICL-D3IE: in-context learning with diverse demonstrations updating for document information extraction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 19485–19494 (2023)
6. Hong, T., Kim, D., Ji, M., Hwang, W., Nam, D., Park, S.: Bros: a pre-trained language model focusing on text and layout for better key information extraction from documents. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 10767–10775 (2022)

7. Huang, Y., Lv, T., Cui, L., Lu, Y., Wei, F.: LayoutLMv3: pre-training for document AI with unified text and image masking. In: Proceedings of the 30th ACM International Conference on Multimedia, pp. 4083–4091 (2022)
8. Jaume, G., Ekenel, H.K., Thiran, J.P.: FUNSD: a dataset for form understanding in noisy scanned documents. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW), vol. 2, pp. 1–6. IEEE (2019)
9. Kenton, J.D.M.W.C., Toutanova, L.K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT, pp. 4171–4186 (2019)
10. Khan, K., Rehman, S.U., Aziz, K., Fong, S., Sarasvady, S.: DBSCAN: past, present and future. In: The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014), pp. 232–238. IEEE (2014)
11. Kuang, J., et al.: Visual information extraction in the wild: practical dataset and end-to-end solution. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) ICDAR 2023. LNCS, vol. 14192, pp. 36–53. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41731-3_3
12. Lewis, D., Agam, G., Argamon, S., Frieder, O., Grossman, D., Heard, J.: Building a test collection for complex document information processing. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 665–666 (2006)
13. Li, C., et al.: StructuralLM: structural pre-training for form understanding. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 6309–6318 (2021)
14. Li, P., et al.: SelfDoc: self-supervised document representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5652–5660 (2021)
15. Li, Y., et al.: Structext: sStructured text understanding with multi-modal transformers. In: Proceedings of the 29th ACM International Conference on Multimedia, pp. 1912–1920 (2021)
16. Liao, H., et al.: Doctr: document transformer for structured information extraction in documents. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 19584–19594 (2023)
17. Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
18. Luo, C., Cheng, C., Zheng, Q., Yao, C.: GeoLayoutLM: geometric pre-training for visual information extraction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7092–7101 (2023)
19. Park, S., et al.: Cord: a consolidated receipt dataset for post-OCR parsing. In: Workshop on Document Intelligence at NeurIPS 2019 (2019)
20. Tang, Z., et al.: Unifying vision, text, and layout for universal document processing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 19254–19264 (2023)
21. Tu, Y., Guo, Y., Chen, H., Tang, J.: Layoutmask: enhance text-layout interaction in multi-modal pre-training for document understanding. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 15200–15212 (2023)
22. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)

23. Wang, D., Ma, Z., Nourbakhsh, A., Gu, K., Shah, S.: DocGraphLM: documental graph language model for information extraction. In: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1944–1948 (2023)
24. Wang, J., Jin, L., Ding, K.: LiLT: a simple yet effective language-independent layout transformer for structured document understanding. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 7747–7757 (2022)
25. Wang, W., et al.: Ernie-mmlayout: multi-grained multimodal transformer for document understanding. arXiv preprint arXiv:2209.08569 (2022)
26. Xu, Y., et al.: Layoutlmv2: multi-modal pre-training for visually-rich document understanding. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 2579–2591 (2021)
27. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: Layoutlm: pre-training of text and layout for document image understanding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1192–1200 (2020)
28. Xu, Y., et al.: LayoutxLM: multimodal pre-training for multilingual visually-rich document understanding. arXiv preprint arXiv:2104.08836 (2021)
29. Xu, Y., et al.: Xfund: a benchmark dataset for multilingual visually rich form understanding. In: Findings of the Association for Computational Linguistics: ACL 2022, pp. 3214–3224 (2022)
30. Zhang, Z., Ma, J., Du, J., Wang, L., Zhang, J.: Multimodal pre-training based on graph attention network for document understanding. IEEE Trans. Multimed. (2022)

Detecting Spiral Text Lines in Aramaic Incantation Bowls

Said Nammneh[1], Boraq Madi[1,4](✉), Nour Atamni[1], Shoshana Boardman[2], Daria Vasyutinsky-Shapira[3], Irina Rabaev[4], Raid Saabni[5], and Jihad El-Sana[1]

[1] Ben-Gurion University of the Negev, Be'er Sheva, Israel
{saeednaa,atamni,el-sana}@post.bgu.ac.il, borak@post.bgu.ac.il
[2] University of Oxford, Oxford, UK
shoshana.boardman@queens.ox.ac.uk
[3] Open University of Israel, Ra'anana, Israel
dariavas@post.bgu.ac.il
[4] Shamoon College of Engineering, Be'er Sheva, Israel
{madibo,irinar}@ac.sce.ac.il
[5] The Academic College of Tel Aviv-Yaffo, Tel Aviv-Yafo, Israel
raidsa@mta.ac.il

Abstract. Due to irregular spacing, character overlap, and varying orientations, text line detection is especially challenging for unconstrained handwritten and historical documents. These complexities make traditional methods, designed for straight lines, insufficient for detecting spiral text lines from Aramaic incantation bowls used in Sasanian Mesopotamia between the 4th and 7th centuries CE. We introduce a novel learning-based method for extracting spiral text lines inscribed on the surfaces of Aramaic incantation bowls. Our model utilizes an encoder-decoder architecture while leveraging connections among corresponding layers, similar to UNet. It combines high-level and low-level features, enabling precise localization and segmenting spiral text lines.

Furthermore, we propose a novel metric to evaluate the dissimilarity between predicted and ground-truth lines. Inspired by the Intersection-over-Union (IoU) metric, We compare our model with the state-of-the-art methods and show that our approach outperforms these methods in terms of the introduced metric.

Keywords: Text Line Detection · Spiral Text Lines · Archaeological Images · Incantation Bowls · UNet

S. Nammneh and B. Madi—Both authors contributed equally to this work.

D. V. Shapira—The participation of Dr. Daria Vasyutinsky Shapira was funded by the European Union (ERC, MiDRASH, Project No. 101071829). Views and opinions expressed are, however, those of the author only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 250–264, 2025.
https://doi.org/10.1007/978-3-031-78495-8_16

1 Introduction

Text Line Detection (TLD) is a fundamental task in the field of historical documents. It remains an active research area in historical document image analysis and handwritten texts, particularly those with irregular spacing, overlapping characters, and diverse orientations. The growing interest is due to the significant impact of text line detection on various tasks, such as text recognition and document categorization. However, TLD can be highly challenging due to the unconstrained nature of handwritten documents, which often feature complex and variable text line layouts, degradation, multiple handwriting styles, scripts, fonts, and other irregularities. These challenges complicate accurately detecting and extracting text lines, requiring sophisticated techniques to handle such documents' diverse and intricate patterns.

During the last decades, numerous methods have been proposed for text line detection and extraction in handwritten documents [2,5,7,10,16]. Existing methods for text line detection, such as [2,18] achieve promising results for horizontal, vertical, and slightly curved lines, they often struggle with more complex patterns like spiral lines, as shown in Fig. 1.

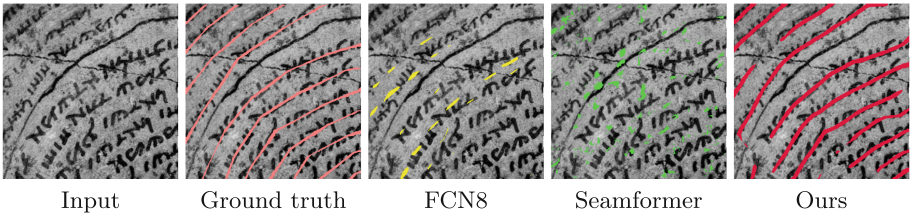


Fig. 1. The performance of our approach compared to two leading methods for detecting spiral text lines on Aramaic incantation bowls. The first and the second images show an input patch and its corresponding ground truth, respectively. The third image shows the results of applying FCN8 [2], where the detecting lines appear in yellow blobs. The result of running Seamformer [18] is presented in the fourth image, where the detecting lines are in green blobs. Seamformer detects blobs and applies post-processing to connect them into polylines. The last image shows the results of our model.

In this paper, we introduce a novel learning-based method for extracting spiral text lines inscribed on the surfaces of Aramaic incantation bowls. These text lines often exhibit perspective distortion in the bowl images. Our model utilizes an encoder-decoder architecture while incorporating connections among corresponding layers, similar to UNet. It combines high-level and low-level features, enabling precise localization and segmentation of text lines. We compare the proposed approach with the state-of-the-art methods and show that it achieved superior accuracy scores.

Our contributions can be summarized as follows:

1. We present a new dataset that includes images of Aramaic incantation bowls and artifacts used in Sasanian Mesopotamia between the 4th and 7th centuries CE [12], and their corresponding ground truth spiral text line annotations, as shown in Fig. 5. These bowls present a unique set of challenges for TLD since they bear inscriptions in various script styles arranged in spiral patterns, reflecting the magical nature of their contents. The images were downloaded from the British Museum collection, and our research team performed the annotation using a specially developed GUI-based annotator, which is freely available to the research community.[1]
2. We develop a novel approach for extracting spiral text lines from Aramaic incantation bowls. Our learning model adopts the encoder-decoder methodology. In addition, we explored various encoder backbones, such as ResNet-18, ResNet-34, ResNet-101, and ResNet-152, as well as advanced residual architectures, such as ResNeXt and WideResNet.
3. We propose two new metrics to evaluate the performance of the models and overcome the sensitivity of Intersection Over Union (IoU) for small and thin objects, named Fuzzy IoU.

The remainder of this paper is organized as follows. Section 2 overviews recent studies and methodologies for text line extraction and detection. Section 3 presents the new dataset and the annotations facilitated by the GUI developed by our team. In Sect. 4, we detail our approach for text-line detection, which utilizes the L-UNet architecture. We present the results of our experiments in Sect. 5, providing quantitative metrics and qualitative analysis of the detection outputs. Finally, in Sect. 6, we conclude with a summary of our findings.

2 Related Work

Text line detection and extraction have long been recognized as a preprocessing step in analyzing handwritten document images. Various methods have been proposed to address this task, primarily focusing on detecting and segmenting horizontal or straight text lines [4,11,17,18]. However, the challenge escalates when dealing with historical documents characterized by irregularities and intricate layouts, such as those found in Aramaic incantation bowls featuring spiral text lines (see Fig. 4). In this section, we review existing methodologies for text line detection and extraction in historical documents, highlighting their relevance and limitations in addressing the unique challenges posed by spiral text patterns.

Droby et al. [4] utilize Mask R-CNN for holistic text line extraction. By training a Mask R-CNN model on document patches, this method effectively detects and segments text lines, achieving state-of-the-art results on well-known datasets of historical documents.

Alaql [1] proposes an approach based on local connectivity maps for text line extraction in historical documents. By utilizing dynamic steerable directional

[1] https://github.com/SaeedYNaa/GUI-For-TLE-data-labeling.

filters, this method effectively estimates the orientation angles of text lines, facilitating accurate segmentation. Furthermore, it introduces adaptive local connectivity maps to address document complexity and low-quality challenges. Rahal et al. [13] presents an approach based on background and foreground information for text line extraction in graphical documents. This method, inspired by the concepts of local connectivity maps, effectively segments text lines while addressing document quality and complexity challenges.

Recent developments in deep learning-based document segmentation have shown promising results in simultaneously addressing various document processing tasks. Oliveira et al. [17] introduce dhSegment, a generic deep-learning approach designed to handle multiple document processing tasks, including page extraction, baseline extraction, layout analysis, and extraction of illustrations and photographs. By employing a CNN-based pixel-wise predictor coupled with task-dependent post-processing blocks, dhSegment offers a flexible and efficient solution capable of handling the diversity of historical document series.

Ronneberge et al. [15] present a language-independent global method for automatic text line extraction from historical document images. The proposed approach computes an energy map of the text image and determines seams that pass across and between text lines. Two algorithms are developed based on this concept, catering to binary and grayscale document images. The algorithms utilize different techniques, such as component extraction and distance transform, to segment text lines accurately. A benchmark dataset containing historical document images with various challenges, including different languages, is also introduced for evaluation purposes.

Next, we provide a detailed overview of fully convolutional deep learning methods categorized as linear-based and UNet-based approaches for text line detection. The approaches represented by architectures such as *FCN* [2] and *Fast and Lightweigh* [11] are considered linear-based methods, while *dhSegments* [17] and *L-UNet* [14] architectures exhibit of UNet-based architecture. Each of these architectures, belonging to both categories, offers unique advantages tailored to various aspects of text line detection, including capturing contextual information and mapping low-resolution feature maps.

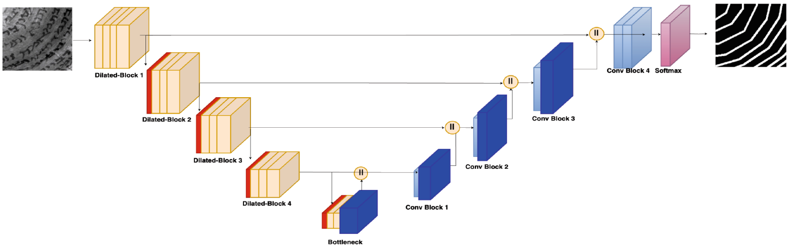


Fig. 2. An illustrative diagram of the UNet architecture, renowned for its skip connections facilitating high-resolution feature maps reconstruction, alongside its contracting (encoder) and expanding (decoder) pathways designed for efficient image segmentation.

2.1 Linear-Based Fully Convolutional Network

This section overviews the linear-based, fully convolutional *FCN* and *Fast and Lightweigh* networks that we utilized for this study.

FCN Architecture. The Fully Convolutional Network (FCN) architecture, depicted in Fig. 3, is based on the FCN proposed for semantic segmentation [2,9]. We experimented with two configurations, FCN8 and FCN32, which comprise an encoder and a decoder. These configurations were selected for their success in page layout analysis of handwritten documents [2]. The encoder downsamples the input image, enabling the filters to capture broader information with a larger receptive field. The decoder combines the final layer of the encoder with lower layers containing finer information and then upsamples the combined layer back to the input size using transpose convolution, where upsampling with a factor f involves applying a convolution filter with a stride equal to $\frac{1}{f}$.

The main difference between FCN8 and FCN32 lies in their levels of upsampling. While FCN32 directly upsamples the final layer of the encoder to the input size, FCN8 integrates the final layer with lower layers and subsequently upsamples the combined layer back to the input size.

Fast and Lightweight Architecture. The proposed network architecture [11] is based on previous research [19], but with modifications to enhance text line detection. It utilizes standard convolutions instead of separable convolutions in early layers and adjusts certain layers while maintaining robust performance. The compact design ensures efficiency with a few trainable parameters and is called from herein as *FastSmall*. The ReLU activation function was employed throughout the network, with sigmoid used in the final layer. An expanded model with small residual connections and extra layers is proposed, referred to here as *FastLarge*.

2.2 UNet-Based Fully Convolutional Network

This section overviews the U-Net-based fully convolutional networks explored in this study.

The **L-UNet** [14] is UNet-based architecture for page layout semantic segmentation and text line segmentation of historical document images. This architecture draws inspiration from the UNet model, particularly Doc-UFCN [3], for its utilization of dilated convolutions, as depicted in Fig. 2. A notable aspect of this architecture is the reduced number of filters, especially in the encoder blocks operating at lower resolutions. In contrast to other architectures that typically double the number of filters at each block level, the L-U-Net architecture argues for a consistent number of filters across all levels, contending that it effectively captures the characteristics necessary for document layout analysis.

The **dhSegment architecture** comprises two main components: an encoding path and a decoding path [17]. Following the ResNet-50 [6] architecture, the encoding path includes five convolutional blocks and progressively reduces the size of feature maps through each step, halving the size at each step. Pretrained weights from ImageNet are utilized to enhance robustness and generalization. The decoding path comprises five blocks and a final convolutional layer, assigning a class to each pixel. Each deconvolutional step involves upscaling the previous block's feature map, concatenating it with the corresponding encoding feature map, and applying a 3 × 3 convolutional layer followed by a ReLU activation. Finally, the output prediction has the same size as the input image, and the number of feature channels constitutes the desired number of classes.

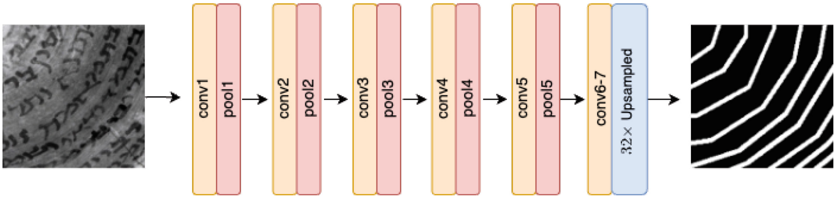


Fig. 3. A Fully Convolutional Network (FCN) architecture. In this example, 32× upsampling is applied to obtain output of the same size as the input, referred as **FCN32**.

3 Dataset

This paper introduces a new dataset to train and evaluate our approach. It contains images of Aramaic incantation bowls downloaded from the British Museum collection. These artifacts, dating back to the Sasanian era (4th to 7th-centuries CE), bear inscriptions characterized by a mixture of Jewish, Zoroastrian, and pagan elements, often arranged in spiral patterns surrounding symbolic images, as shown in Fig. 4. The inscriptions typically consist of incantations, prayers, and curses, reflecting the syncretic religious practices prevalent.

To facilitate the extraction of spiral text lines from the bowl images, we annotate the dataset using a specially developed annotation tool that includes an intuitive graphical user interface (GUI). It enables marking the location of text lines within the bowl images.

We generate a polyline list and a mask for each image in the dataset. The polyline list indicates the center of the text lines, and the mask image delineates the text line regions using white spiral lines against a black background. This annotated dataset is of great value for developing and evaluating methodologies aimed at extracting text lines from spiral patterns.

Figure 5 illustrates the data labeling and clarifies converting the raw image into a labeled data tuple containing the original image and the corresponding labeled mask.

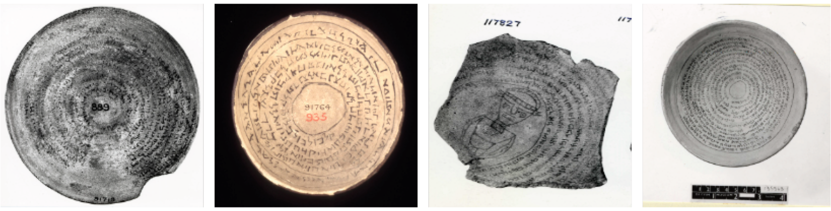


Fig. 4. Samples of the Aramaic incantation bowls images.

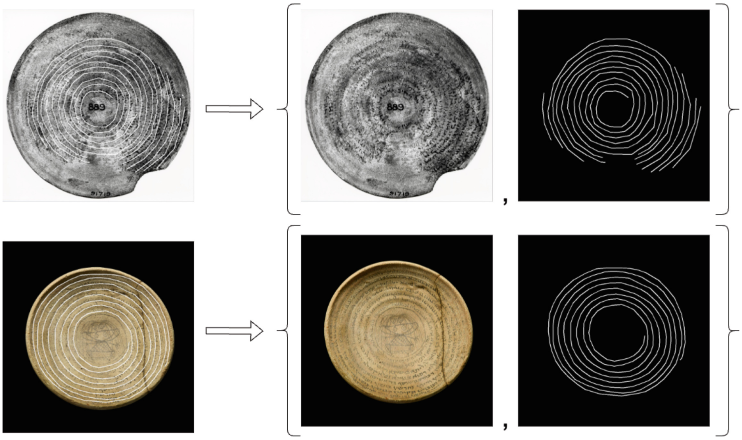


Fig. 5. The image on the left shows the original bowl image and the annotation (overlay) as it is displayed in our annotation tool. The two images on the right show the original image and the annotated mask, displayed in greyscale, which delimits the lines of text in the original image.

4 Method

We present a novel learning-based approach for extracting spiral text lines inscribed on Aramaic incantation bowls' surfaces. These text lines are also subject to perspective distortion as we process images of these bowls. Our model follows the encoder-decoder methodology, which includes Fully Convolutional Networks and U-Net.

The encoder-decoder architecture consists of an encoder that compresses the input image into a lower-dimensional representation through a series of convolutional layers and downsampling operations, capturing essential features while reducing spatial dimensions. The decoder reconstructs the compressed representation into the original spatial dimensions using upsampling operations and convolutional layers, restoring image details and resolution.

UNet, a type of encoder-decoder architecture specifically designed for image segmentation tasks, features a symmetric structure with a contracting path (encoder) and an expansive path (decoder), incorporating skip connections between corresponding layers to retain fine-grained spatial information, as shown in Fig. 2.

Our model is used for pixel-wise predictions, classifying each pixel in the input image into categories such as background or text. It incorporates connections among corresponding layers, similar to U-Net, and combines high-level and low-level features, enabling precise localization and segmentation of the text lines, even with perspective distortions. Our model adopts the ResNeXt as a backbone for encoding.

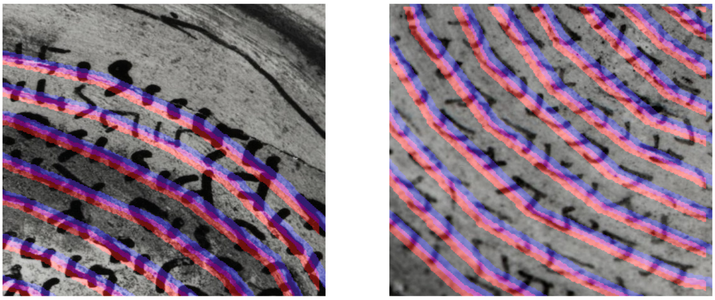
The encoding path starts with the initial layer, including the 7 × 7 convolutional layer, followed by BatchNorm, ReLU activation, and Max pooling; then, it consists of 4 layers whose content varies according to the backbone. Each layer includes multiple convolutional layers, batch norms, and activation functions but excludes max pooling layers. It is structured to progressively reduce the spatial dimensions of the input while increasing the depth (number of filters), allowing the network to capture complex features at multiple scales. These features typically include edges, textures, and complex patterns essential for accurately detecting and extracting text lines.

The decoding path comprises five blocks and a final convolutional layer, which assigns a class to each pixel. During each step in the decoding path, the feature map from the previous block is upscaled, concatenated with the corresponding feature map from the encoding path, and processed through a 3 × 3 convolutional layer followed by a ReLU activation. This concatenation ensures that spatial information from the encoding path is preserved and utilized effectively. The output prediction retains the same dimensions as the input image, with the number of channels corresponding to the desired number of classes, allowing for precise segmentation of the spiral text lines from the background and other elements in the bowls.

4.1 Fuzzy IOU

Intersection over Union (IoU) is an evaluation metric extensively employed across computer vision tasks to assess the precision of object detection algorithms. It was initially introduced to compare bounding boxes, but it has been applied to evaluate mask segments, particularly in semantic and instance segmentation tasks. Its effectiveness dwindles when applied to small objects such as text lines, primarily due to their diminutive size. Though seemingly insignificant at first glance, factors like subtle translations, rotations, or resizing of predicted text lines can unexpectedly wield a disproportionate influence on resulting IoU scores. These minor alterations may not appear critical to the overall outcome, and they might be deemed acceptable in the results, as shown in Fig. 6. However, when considering IoU values, they fail to reflect this lack of criticality, potentially leading to significant distortions in assessing model performance.

To address the limitation of IoU, we augment the thickness of text line annotations (the polylines) to diminish the metric's vulnerability to minor alterations like small translations and adjustments. This augmentation enhances the evaluation process's resilience.



(a) (b)

Fig. 6. Results of text line detection. The red and blue lines represent the model predictions and the ground truth, respectively. Despite the noticeable visual disparity, the difference between the blue (ground truth) and red (prediction) lines is not critical. In (a), there is an average translation of 5 pixels in the x-direction, whereas in (b), the translation is 5 pixels in the y-direction. (Color figure online)

We propose two approaches for augmenting the thickness of text line annotations (polylines). The first one avoids penalization for small translations and is considered fully correct results. It extends the annotation boundaries by t pixels on each side, making the lines more thicker, as shown in Fig. 7b. This is achieved by applying a kernel of size $t \times t$ using morphological dilation operations.

The second aims to handle minor translations and adjustments based on a fuzzy evaluation. It involves augmenting the polylines with weights that vary as we move away from the center, as depicted in Fig. 7c. These weights are calculated based on the Distance Transform with Euclidean distance, normalized between 1 at the center of the text line and 0 at the outer edges.

Let's define P as the set of predicted text lines and G as the set of ground truth text lines (polylines). Additionally, G_t denotes the ground truth after augmentation with t pixels using the dilation operator, where t represents the expansion parameter. We calculate the fuzzy Intersection over Union (IoU_t) as the intersection of P with G_t, divided by the union of P and G, as shown in Eq. 1.

$$IoU_t = \frac{P \cap G_t}{P \cup G} \tag{1}$$

G_d represents the ground truth after augmentation with the Distance Transform and using Euclidean distance. Similar to Eq. 1, we define the IoU metric (IoU_d) to be the intersection of P with G_d, divided by the union of P and G. This approach comprehensively assesses text line detection performance under different augmentation techniques.

$$IoU_d = \frac{P \cap G_d}{P \cup G} \tag{2}$$

The modified IoU metrics, IoU_t and IoU_d, offer a nuanced assessment of text line detection systems by accounting for text line thickness augmentations vari-

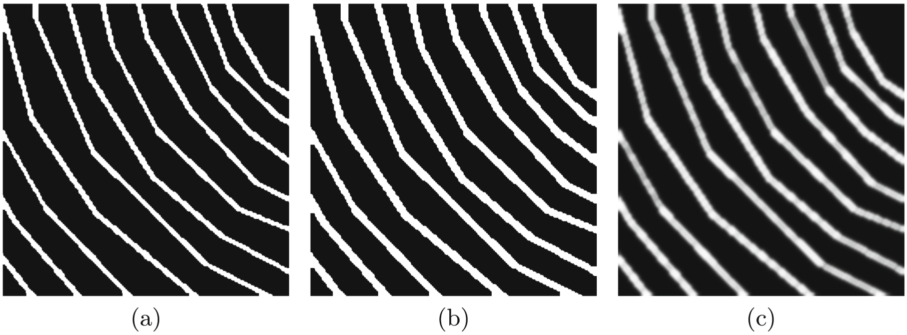


Fig. 7. (a) Ground Truth of Text Lines: Displays the true representation of text lines. (b) Expansion Using Dilation: Shows dilation-based expansion, with minor alterations considered correct. Thickness increases by t pixels on each side. (c) Weighted Results: Illustrates weighted augmentation of text lines, with weights varying gradually from 1 at the center to 0 at the edges, calculated using Distance Transform with Euclidean distance normalization.

ations. They evaluate how accurately detected text lines align with the ground truth under the specified augmentation technique. Our choice of the thickening augmentation technique aims to balance sensitivity to variations in text line predictions with resilience against minor perturbations. This enhances the stability and reliability of model evaluations, providing clearer insights into its performance in text line detection tasks.

5 Experimental Evaluation

We have experimentally evaluated our model and compared its performance with the state-of-the-art methods. Initially, we employed linear FCN models, including FCN8, FCN32, FastSmall and FastLarge [11] to detect spiral text lines. However, these models struggled to grasp the complexity of spiral text line detection, leading to unsatisfactory results, as shown in Fig. 1.

Therefore, our evaluation focused on U-Net-based models, which can effectively capture the low-level and semantic features. The L-UNet, which is a lightweight yet robust model equipped with dilatation convolution blocks, exhibits notable improvements over linear models, attributing its success to the integration of residual/skip connections between encoder and decoder blocks, facilitating the fusion of features essential for accurate text line detection.

The dhSegment, an UNet-based model fortified with residual dilatation convolution blocks and adopts ResNet50 as its encoder, demonstrates superior capabilities compared to L-UNet. Still it does not output accurate results compared to our model, as shown in Table 1.

We explore various UNet architecture encoding path backbones, including ResNet-18, ResNet-34, ResNet-101, and ResNet-152. In addition, we examined advanced residual techniques such as WideResNet and compared their performance with our model (See Sect. 4).

5.1 Dataset Preparation

The dataset preparation involves extracting patches of size 512 × 512 that capture a significant portion of a bowl image. We ensure that each patch contains multiple lines of text to provide enough spatial information for the model to learn the intricate patterns and variations present in the text lines.

To generate patches, we apply a sliding window to the original images using a stride of 256 pixel in both X and Y directions, creating an overlap among consecutive windows. This approach ensures comprehensive coverage of the bowl images while maintaining contextual information across adjacent patches.

We avoid generating patches that do not include text lines by utilizing the annotated text lines to guide the calculation of tight bounding boxes (include text) and applying the sliding window within these boxes. Similarly, the annotated data (the corresponding masks) undergoes the same bounding process, ensuring that each patch of the bowl image corresponds to its respective patch label as illustrated in Fig. 8.

The original bowl images are split into three sets with a ratio of $70\%, 15\%$, and 15% for the train, validation, and test subsets, respectively. We allow an overlap of bowls between the train and validation sets.

5.2 Training

Throughout the training process, for all the models, we utilize the Adam optimizer [8], alongside a batch size of 8 for a duration of 30 epochs, training each model from scratch with a learning rate of 0.002. The training used patches of size 512 × 512, where 0 denoted the background and 1 denoted the text line region. Additionally, binary Cross Entropy was employed as the loss function to minimize the loss during the training.

5.3 Results and Analysis

The overlapping patches of size 512 × 512 are extracted from the test set images during the inference stage. We assessed the results using the two approaches outlined in Sect. 4.1. For the IoU_t metric, we varied the expansion parameter t, considering values of 2, 3, 4, and 5 as detailed in Table 1. These variations are labeled as IoU_2, IoU_3, IoU_4, and IoU_5, respectively. Additionally, we conducted another evaluation to calculate the models' performance using the IoU_d metric, as presented in Table 1.

Table 1 shows varying performance metrics across different models and evaluation techniques. For instance, models FCN32 and FCN8 exhibit lower scores than the ResNet101, ResNet18, ResNet34, and ResNet50 models. Among these,

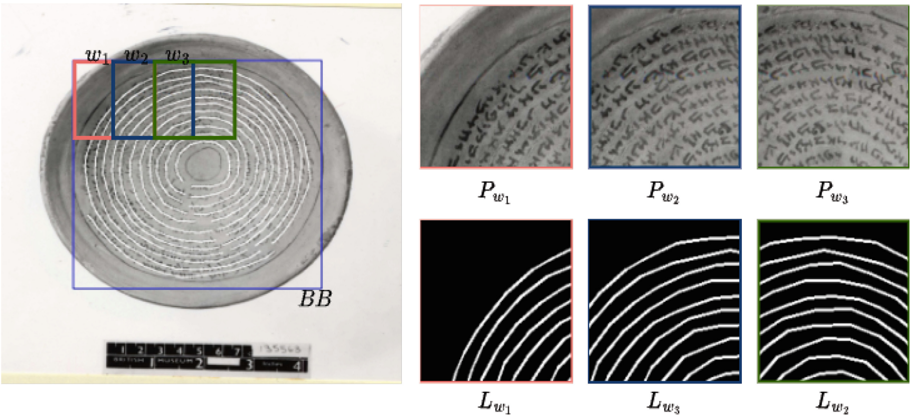


Fig. 8. The result of the patching techniques applied to the original image, where the leftmost panel illustrates the annotated image alongside the bounding box encapsulating the annotated region and three overlapping sliding windows $\{w_i\}_{i=1}^{3}$ applied to the image. Morphological operations, specifically dilation, ensure a slight margin between the bounding box and the outermost spiral line. This operation increases the annotation lines' thickness, creating space between the original annotation lines (text) and the bounding box covering the largest contours of the dilated annotation image. The right part illustrates the resulting extracted patch images p_{w_i} and their corresponding annotation patch labels L_{w_i}.

our method with ResNext101_32x8d and ResNext50_32x4d as the encoding backbone, as shown in Fig. 9, stand out with the highest scores among other models, respectively, for the IoU_t and IoU_d metrics.

ResNext models demonstrate superior performance to WideResNet and standard ResNet due to their enhanced architecture, incorporating a more efficient feature extraction and representation learning design. This enhanced architecture allows ResNext models to capture more diverse and complex patterns within the data, leading to improved accuracy and performance in various tasks.

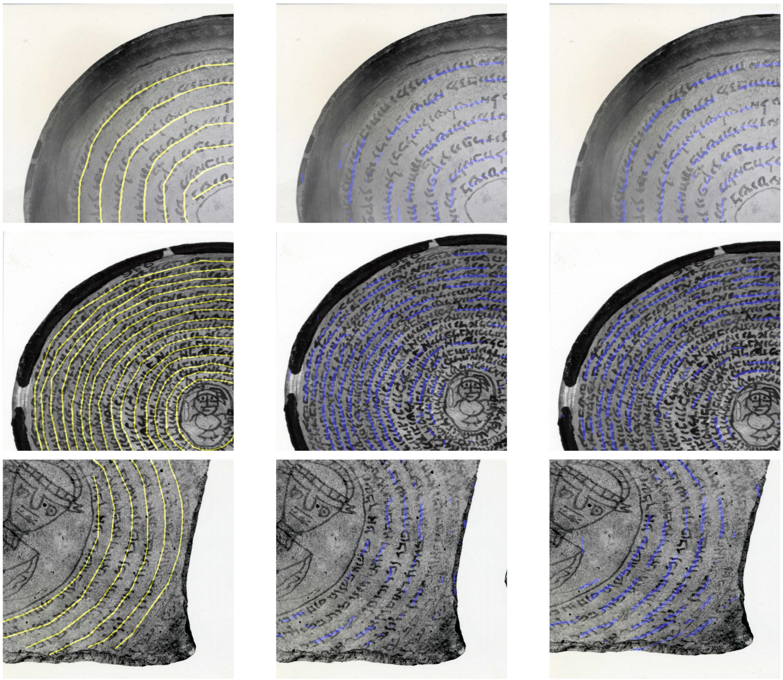
The lower values of IoU_d compared to the IoU_t reflect the differences in the way the two methods handle augmentation of text line annotations(polylines). While dilation uniformly expands the predicted text lines, the distance transform method assigns varying weights to pixels based on their distance from the text line center. This nuanced approach may result in some predicted pixels being assigned lower weights, particularly those farther away from the boundary.

6 Conclusions

In this study, we have introduced a novel method for detecting and extracting spiral text lines by utilizing an encoder-decoder similar to UNet architecture, which combines high-level and low-level features for precise segmentation. To facilitate the experiments, we created a GUI-based annotator to annotate the

Table 1. Model Results: Performance metrics of different models using IoU_t and IoU_d techniques.

Model	IoU_2	IoU_3	IoU_4	IoU_5	IoU_d
FCN32	17.97	19.40	20.84	22.22	19.12
FCN8	17.92	19.35	20.77	22.15	12.86
L-UNet	70.40	72.81	74.4	74.74	51.12
ResNet101	88.45	90.75	92.04	91.48	71.18
ResNet18	72.38	74.44	75.78	76.80	58.54
ResNet34	82.93	84.33	85.25	85.60	67.18
dhsegment	74.93	76.24	77.94	78.37	59.88
Ours w/ResNext101	**89.95**	90.41	91.39	91.53	**75.24**
Ours w/ResNext50	89.81	**91.20**	**92.22**	**92.41**	74.69
WideResNet101	89.44	91.09	92.11	92.34	73.86
WideResNet50	87.35	88.42	89.18	89.30	72.88



$GroundTruth$ $ResNext50$ $ResNext101$

Fig. 9. Series of images showing the ground truth alongside predictions from two different architectures in our experiment: $ResNext50$, and $ResNext101$.

dataset, easily creating plenty of datasets for the learning process. In addition, we have shown that training linear-based FCN and U-Net-based architectures for detecting spiral text lines on Aramaic incantation bowls is possible. We also showed that the UNet-based models show superior performance compared to linear-based models.

We have demonstrated that the Intersection over Union (IoU) metric has limitations when used for spiral line detection. To address these limitations, we introduced the fuzzy IoU metric that avoids penalization for small translations between the ground truth and the predicted polylines by applying morphological operation and calculating the distance transform map of the ground truth polylines.

The scope of future work focuses on exploring the applicability of our approach to other datasets that include highly curved or circular text lines.

References

1. Alaql, O.: Text line extraction for historical documents images using local connectivity map. Ph.D. thesis, Kent State University (2014)
2. Barakat, B.K., Droby, A., Alaasam, R., Madi, B., Rabaev, I., El-Sana, J.: Text line extraction using fully convolutional network and energy minimization. CoRR, abs/2101.07370 (2021)
3. Boillet, M., Kermorvant, C., Paquet, T.: Multiple document datasets pre-training improves text line detection with deep neural networks. In: 2020 25th International Conference on Pattern Recognition (ICPR). IEEE (2021)
4. Droby, A., Barakat, B.K., Alaasam, R., Madi, B., Rabaev, I., El-Sana, J.: Text line extraction in historical documents using mask R-CNN. Signals **3**(3), 535–549 (2022)
5. Grüning, T., Leifert, G., Strauß, T., Labahn, R.: A two-stage method for text line detection in historical documents. CoRR, abs/1802.03345 (2018)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR, abs/1512.03385 (2015)
7. Huang, Z., Gu, J., Meng, G., Pan, C.: Text line extraction of curved document images using hybrid metric. In: 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), pp. 251–255 (2015)
8. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
9. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. CoRR, abs/1411.4038 (2014)
10. Louloudis, G., Gatos, B., Pratikakis, I., Halatsis, C.: Text line detection in handwritten documents. Pattern Recogn. **41**(12), 3758–3772 (2008)
11. Melnikov, A., Zagaynov, I.: Fast and lightweight text line detection on historical documents. In: Bai, X., Karatzas, D., Lopresti, D. (eds.) DAS 2020. LNCS, vol. 12116, pp. 441–450. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-57058-3_31
12. Naamneh, S., et al.: Classifying the scripts of Aramaic incantation bowls. In: Proceedings of the 7th International Workshop on Historical Document Imaging and Processing, HIP 2023, pp. 55–60 (2023)

13. Roy, P.P., Pal, U., Lladós, J.: Text line extraction in graphical documents using background and foreground information, pp. 227–241 (2012)
14. Rahal, N., Vögtlin, L., Ingold, R.: Layout analysis of historical document images using a light fully convolutional network. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) ICDAR 2023. LNCS, vol. 14191, pp. 325–341. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41734-4_20
15. Saabni, R., Asi, A., El-Sana, J.: Text line extraction for historical document images, pp. 23–33 (2014)
16. Shejwal, M.A., Bharkad, S.D.: Segmentation and extraction of text from curved text lines using image processing approach. In: 2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC), pp. 1–5 (2017)
17. Oliveira, S.A., Seguin, B., Kaplan, F.: dhSegment: a generic deep-learning approach for document segmentation (2019)
18. Vadlamudi, N., Krishna, R., Sarvadevabhatla, R.K.: SeamFormer: high precision text line segmentation for handwritten documents. In: Fink, G.A., Jain, R., Kise, K., Zanibbi, R. (eds.) ICDAR 2023. LNCS, vol. 14190, pp. 313–331. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41685-9_20
19. Zharkov, A., Zagaynov, I.: Universal barcode detector via semantic segmentation. CoRR, abs/1906.06281 (2019)

Incorporating Domain Knowledge in Multi-objective Optimization Framework for Automating Indian Legal Case Summarization

Shreya Goswami[1], Naveen Saini[2](✉), and Saurabh Shukla[1]

[1] Indian Institute of Information Technology Lucknow, Lucknow, India
{mcs21023,saurabh.shukla}@iiitl.ac.in
[2] Indian Institute of Information Technology Allahabad, Prayagraj, India
nsaini@iiita.ac.in

Abstract. Legal cases can be lengthy and complex, often containing much information, evidence, and legal arguments. On the contrary, attorneys, judges, and legal researchers must review multiple cases to build arguments, make decisions, or conduct legal research. Summarizing cases can help legal professionals and researchers quickly grasp the essential details without reading through every word of the original documents. While various legal case summarization algorithms exist in the literature, they lack a systematic integration of the various summary quality factors necessary to create comprehensive and concise legal summaries. To address this gap, the present paper proposes an innovative extractive summarization approach by leveraging the use of legal-domain knowledge in the multi-objective optimization-based evolutionary framework. Our method simultaneously optimizes different objectives, including legal domain knowledge, tf-idf scores, and diversity, to get good-quality summaries. As per our knowledge, this work is the first of its kind in utilizing the efficacy of a multi-objective evolutionary algorithm for generating legal document summaries. For evaluation, we thoroughly conduct experiments on legal case documents from the Indian Supreme Court, accompanied by gold-standard summaries created by legal experts. The results obtained reveal that our algorithm demonstrates significant improvements of 16.00%, 12.91%, 13.65%, and 0.18%, over the state-of-the-art technique, in terms of ROUGE-2 F1, ROUGE-L F1, ROUGE-2 Recall, and ROUGE-L Recall, respectively. Further, the statistical significance of the results is also validated.

Keywords: Legal Document · Summarization · Multi-objective Optimization · Evolutionary Algorithm · Domain Knowledge

1 Introduction

Long and complicated court cases can involve an extensive amount of data, supporting documentation, and legal reasoning. Quite the opposite; in order to

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 265–280, 2025.
https://doi.org/10.1007/978-3-031-78495-8_17

figure out their points of view, reach judgments, or carry out legal study, lawyers, judges, and legal researchers must investigate a number of instances. In these scnerio, legal case summarization significantly enhances the efficiency and accessibility of the legal system. By distilling intricate case details into key points, these summaries save time for legal professionals, such as lawyers, judges, and researchers, who need to grasp case complexities quickly. They also democratize legal information, making it accessible to non-lawyers, including clients and the general public. Summaries serve as quick reference guides for professionals managing multiple cases and simplify legal research by highlighting precedent-setting cases. In legal education, they aid in teaching significant cases and principles. Additionally, summaries expedite decision-making processes by providing a clear overview of cases, facilitating informed choices about legal strategies and resource allocation. Document summarization is commonly divided into two main categories: abstractive and extractive. The first one involves rephrasing and synthesizing information as shown by studies conducted by Liu et al. [12] and Nallapati et al. [18]. On the other hand, in *extractive* Summarization, key sentences are chosen from the original document and incorporated into the summary. Examples of extractive approach include works such as LexRank [7], Gist [11], and Luhn [13]. This paper focuses primarily on extractive summarization, given its prevalent use in the context of legal case documents [1].

Challenges: In the legal field, understanding and summarizing complex legal documents pose significant challenges. These documents often contain difficult language and specific legal terms, requiring expertise in law to interpret accurately. Additionally, legal case files can be lengthy, spanning many pages, making manual review time-consuming and resource-intensive. Furthermore, the structures of legal cases vary widely, making it hard to identify essential points for summarization. Ensuring that summaries accurately capture the legal context, including facts, arguments, and court decisions, is crucial to avoid misunderstandings. Summarization also involves subjective choices, which can introduce bias if not carefully managed. Despite these challenges, the development of automatic summarization systems for legal documents continues to be a practical and valuable pursuit.

Objective of the Paper: This paper aims to introduce an innovative method for summarizing Indian Legal Case documents. It addresses the need for enhanced summarization techniques in the legal domain, especially concerning cases from the Indian Supreme Court. The central objective is to present a multi-objective optimization algorithm capable of effectively considering and integrating various factors to produce summaries of superior quality. Through a series of comprehensive experiments conducted on Indian case documents sourced directly from the Indian Supreme Court, this study endeavors to showcase the efficacy of the proposed algorithm. Furthermore, it seeks to conduct a thorough comparison between the performance of proposed algorithm and existing benchmarks. By tackling the inherent challenges associated with legal document summarization, including intricate language and diverse document structures, the paper endeavors to make a significant contribution to enhancing the efficiency and accessibility of the legal system. Thereby, benefiting legal professionals and researchers alike. Various scoring features/objective functions such as the presence of legal terminology, highest tf-idf score per sentence, and anti-redundancy measures are concurrently optimized utilizing the multiobjective binary differential evolution algorithm [22]. The first objective is achieved by incorporating domain knowledge related to legal domain which consist of a dictionary of legal catchphrases. The second objective is based on well-known term frequency and inverse document frequency concept in the information retrieval, while third objective is based on maintaining diversity among sentences in the summary.

Major Contributions: This paper's primary contributions include: (a) Development of a multi-objective evolutionary optimization framework capable of integrating various factors to produce high-quality summaries for Indian Legal Cases; (b) Along with tf-idf score and anti-redundancy measure, using Indian Legal Catchphrases (Catchwords[1]) as one of the objectives for creating legal summaries which helps in incorporating domain knowledge; (c) Demonstration of the efficacy of the proposed algorithm through experiments which are conducted comprehensively on Indian Supreme Court case documents, providing empirical evidence of its effectiveness; (d) Conducting a thorough comparison with existing benchmarks, including supervised and unsupervised summarization models.

For experimentation, we will utilize the dataset employed by DELSumm [2]. The dataset comprises 50 legal case documents from the Indian Supreme Court. For evaluation, we rely on gold standard summaries provided by two experienced law scholars, each corresponding to a document within the dataset. The results are evaluated using well-known ROUGE scores.

This paper is organized as follows: The literature survey is described in Sect. 2. Section 3 states the problem statement. Section 4 explains the details of the method that is proposed. Section 5 has the experimental setup and Sect. 6 has the discussion on the results obtained. Lastly, the conclusion is presented in Sect. 7.

2 Related Works

Extractive text summarization techniques are aimed at identifying crucial sentences from a document to form a summary. In literature, various methods have been developed for legal document summarization which falls into four main categories: (a) Supervised domain-specific; (b) Unsupervised domain-specific; (c) Supervised domain-independent; (d) Unsupervised domain-independent. An example of *supervised domain-specific* methods is *Gist* by Liu et al. [11]. Gist extract sentences using features such as word count and sentence position. It applies machine learning classifiers such as Decision Tree, MLP, and LSTM to obtain the final summary. However, it was designed and tested for Chinese legal case documents, thus limiting its applicability to other contexts. *Unsupervised*

[1] For more information, see https://irwinlaw.com/cold/catchwords.

domain-specific methods include tailored approaches for legal case documents. For instance, LetSum [8] considers rhetorical structure and cue-phrases to assign roles to sentences, while in [24], Saravanan et al. use a K-Mixture Model. Another method, CaseSummarizer [20] evaluates sentence importance based on factors like TF-IDF values and presence of dates or named entities. Zhong et al. [26] utilized Maximum Margin Relevance (MMR) to select sentences from the legal document. While CaseSummarizer does not consider rhetorical roles, but LetSum, KMM, and MMR incorporate them post-summarization.

Supervised domain-independent methods treat extractive text summarization as a problem of binary categorization, with sentence representations learned using a hierarchical encoding method. Methods like NeuralSum [3] and SummaRunner [18] employ Recurrent Neural Network (RNN) encoders to understand sentence representations and select sentences for inclusion in the summary based on factors like content, salience, novelty, and position importance. Recently, pretrained encoders like BERT [5] have gained popularity for their ability to directly output sentence representations, which can be fine-tuned on domain-specific data for supervised summarization tasks. In *unsupervised domain-independent* approaches, the commonly used algorithms employ Frequency-based methods such as Luhn [13], graph-based methods such as LexRank [7], to identify important sentences where the summary comprises of the best-ranked sentences.

Findings from Literature: From the literature, we have found that multi-objective evolutionary algorithm (MEA) [27] have shown their advancement over different summarization tasks like microblog summarization [22], figure summarization [23], bug report summarization [17], bio-medical article summarization [14]. But the same has not been adopted legal document summarization. Motivated by this, we have adopted the same in our proposed framework. The concept of *Multi-objective optimization* (MOO) aims at concurrently optimizing multiple objective functions, while evolutionary algorithm is inspired by biological phenomenon of the human beings. At the end of execution, MEA presents a collection of alternative solutions known as the Pareto optimal set to the decision maker. In essence, an MOO problem can be represented as:

$$\text{maximize } \{f_1(\tilde{\mathbf{v}}), f_2(\tilde{\mathbf{v}}), \ldots, f_m(\tilde{\mathbf{v}})\} \text{ such that } \tilde{\mathbf{v}} \in V \tag{1}$$

$V = \{\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \ldots, \tilde{\mathbf{v}}_n\}$ represents a feasible set of decision vectors in an n-dimensional space. Here, $m \geq 2$ denotes the number of objective functions to be maximized. Constraints may also be incorporated into the optimization process to address specific requirements or limitations.

In [22], Saini et al. applies multi-objective optimization in the context of microblog/tweet summarization, aiming to generate summaries of good quality. Authors employs various statistical quality measures such as tf-idf score, tweet length, and anti-redundancy, which are concurrently optimized using a multi-objective differential evolution technique. However, in our investigation, we will focus on utilizing tf-idf score, the frequency of Indian Legal catchphrases, and anti-redundancy as our objectives to produce high-quality Indian legal summaries. For incorporating *domain knowledge* in our framework, we have experimented by incorporating the presence of legal catchphrases while creating the summaries and thus, serving as one of the objectives. The work done in [15] is used to automatically identify catchphrases from legal court case documents. It mainly focuses on legal terms and noun phrases. In this method, legal catchwords/catchphrases are generated for each document; therefore, it provides an extensive list of all the legal terms and noun phrases specifically present in each legal document. *Word Mover Distance (WMD)* [10] is employed to assess the dissimilarity between two sentences. It does so by quantifying the distance that the embedded words [16] of one sentence must traverse to match those of another sentence [10]. In our methodology, each text represents a sentence. Word embeddings for various words are derived using InLegalBERT and InCaseLawBERT [19] models. When two sentences demonstrate similarity, their associated WMD value tends to approach 0. With the help of WMD, we have tried to achieve anti-redundancy which is considered as another objective in our approach. *Term Frequency-Inverse Document Frequency* (TF-IDF) [25] is a technique for assessing the importance of a term in a document with respect to a collection of documents. It aims to emphasize on terms that are both frequently present within a document and are rare across a collection of documents. This helps in capturing the significance of a term in a specific context. For our case, we have taken maximum tf-idf scores as one of our objectives.

3 Problem Definition

Consider an Indian Legal Case Document D consisting of N sentences, represented as $D = \{s_1, s_2, \ldots, s_N\}$. Our objective is to identify a subset of sentences within a maximum and minimum word count, comprising one-third of D, denoted as $T \subseteq D$, that satisfies the following constraints:

$$\frac{\sum_{i=1}^{N} \text{wordCount}_i}{3} \leq \sum_{i=1}^{N} B_i \leq \frac{\sum_{i=1}^{N} \text{wordCount}_i}{2} \tag{2}$$

where, $\sum_{i=1}^{N} wordCount_i$ represents the total word count of the document, and N denotes the total number of sentences in D, B_i represents a binary variable indicating whether sentence s_i is included in the summary T with $B_i = 1$ if $s_i \in T$; otherwise $B_i = 0$. Our goal is to maximize the following set of objective functions: $\max\{\vartheta_1(T), \vartheta_2(T), \vartheta_3(T)\}$, where ϑ_1, ϑ_2, and ϑ_3 are the objective functions described in the forthcoming sections. The optimization problem can involve either two or three objective functions, depending on the specific requirements. These objective functions quantify different aspects of the summaries, collectively enhancing the quality of the generated summary. This multi-objective optimization concept is integrated with the evolutionary framework which starts from a set of candidate solutions, each representing a summary. In other words, each solution in the population represents a subset of sentences which is evaluated based on the mentioned objective functions within a multi-objective optimization (MOO) framework and continuously optimized using the iterative procedure of evolutionary algorithm. For our purpose, we have utilized the efficacy of multi-objective binary differential evolution algorithm [22]. The description of the objectives functions used in the proposed algorithm, is discussed below:

3.1 Maximum TF-IDF Score of the Sentences (ϑ_1)

The term frequency-inverse document frequency (TF-IDF) measure is utilized to allocate weights to different words in the sentences and should be maximized. It is expressed as:

$$\vartheta_1 = \frac{1}{|T|} \sum_{i=1}^{|T|} \sum_{\text{word}k \in s_i, s_i \in T} wk, s_i \tag{3}$$

where w_{k,s_i} represents the TF-IDF score of the k-th word (word_k) present in the i-th sentence s_i of summary T, and $|T|$ is the total number of sentences in the summary.

3.2 Maximum Number of Legal Catchphrases (ϑ_2)

Objective function ϑ_2 considers the maximum count of legal catchphrases in the summary. Increasing the number of legal catchphrases means including more legal information, which is often important. It is expressed as:

$$\vartheta_2 = \sum_{i=1}^{|T|} \text{numberOfLegalCatchphrases}(s_i) \tag{4}$$

where numberOfLegalCatchphrases(s_i) denotes the number of legal catchphrases in the i-th sentence in the summary T after removing stop words. To get the list of catchphrases, the work done in [15] is utilized which automatically identify catchphrases from legal court case documents.

3.3 Anti-redundancy (ϑ_3)

To minimize redundancy in the summary, we introduce the objective function ϑ_3. Each sentence is compared with every other sentence. It is calculated as the sum of WMD between each pair of sentences. It is expressed as:

$$\vartheta_3 = \frac{1}{|T|} \sum_{i,j=1, i \neq j}^{|T|} wordMoverDistance(s_i, s_j) \tag{5}$$

where s_i and s_j represent the i-th and j-th sentences in the summary T respectively, and $|T|$ is the total number of sentences in the summary.

4 Proposed Methodology

This paper introduces an extractive Indian Legal Case summarization system. It employs a binary differential evolution technique in integratiin with multi-objective optimization. It has the following phases:

4.1 Solution Representation and Population Initialization

In evolutionary algorithms, a population of solutions, denoted as P, comprises solutions (also known as chromosomes), represented by binary vectors $\{\tilde{\mathbf{v}}_1, \tilde{\mathbf{v}}_2, \tilde{\mathbf{v}}_3, \tilde{\mathbf{v}}_4, \ldots, \tilde{\mathbf{v}}_{|P|}\}$. Each solution corresponds to a set of sentences in the dataset being summarized which in turn should contain a certain number of words. If the dataset contains N sentences, each with an average of $\frac{n}{N}$ words per sentence, then the length of each solution vector is N. For example, if an event/document comprises 6 sentences, an acceptable solution could be represented as [1, 0, 1, 0, 0, 1], indicating that the first, third, and sixth sentences from the original event are included in the summary. After selecting the sentences, we also check whether it falls within the maximum and the minimum words range to satisfy the summary length constraint. The initial population is generated by shuffling the sentences randomly and considering the number of words in each solution varying between a minimum ($S_{\min}$) and maximum ($S_{\max}$) threshold. Here, $S_{\min}$ and $S_{\max}$ represents minimum and the maximum number of words that should be present in a summary. According to [2], the preferred length of legal document summaries should be one-third the length of the document. Thus, the word count of our summaries should be approximately one-third of the word count of the document.

4.2 The Objective Functions Calculation

To craft a high-quality summary, identifying appropriate objective functions or quality metrics is imperative. These functions assess the content of the subset of sentences included in a solution. This helps in attaining summaries which are comprehensive and yet concise. All such objective functions have been thoroughly elaborated in Sect. 3 and are uniformly oriented towards maximization.

4.3 The Genetic Operators

The process of generating new solutions from existing ones in the differential evolutionary framework involves three important steps: mating pool generation, crossover, and mutation. These steps collectively contribute to the creation of a new population denoted as P'. The steps are detailed below:

(a) Generation of Mating Pool: The mating pool consists of solutions capable of producing offspring through mating. To construct the mating pool for a given solution (called as current solution), we consider only neighboring solutions identified through a neighborhood mechanism. Both exploration and exploitation behaviors are taken into account during the pool generation. The steps are as follows:

1. Identify the solutions in the vicinity of the current solution.
2. Generate a random probability p.
3. If $p < \beta$, include a subset of neighboring solutions in the mating pool to encourage exploitation.

4. If $p \geq \beta$, include all neighboring solutions in the mating pool to encourage exploration.

(b) Mutation: For mutation, first probability estimation operator (PEO) is applied between the randomly selection solutions from the mating pool and the current solutions. Thereafter, the obtained probability values corresponding to the different components of the current solution are converted to binary values based on some heuristic. PEO is defined as:

$$P(v_t^j) = \frac{1}{1 + e^{-\frac{2b \times [v_t^{p1,j} + F \times (v_t^{p2,j} - v_t^{p3,j}) - 0.5]}{1+2F}}}$$

where, v^{p1}, v^{p2}, v^{p3} are the three randomly selected solutions, $v^{pi,j}$ represents the j-th component of i-th random solution, b and F are the real positive constant and scaling factor in differential evolutionary procedure, t denote the generation number. For more detail about it and its conversion to binary vectore, reader can refer to [21].

(c) Crossover: The crossover operation involves exchanging components between the current solution and the binary vector obtained in the mutation. This results in the generation of a new solution. For Eq. involved, refer to [21].

4.4 Selection and Termination

The previous population P is merged with the newly generated population P'. Subsequently, the best solutions (top $|P|$) are selected for the next generation based on non-dominated sorting and crowding distance operators of NSGA-II [4] algorithm. This process continues until a maximum number of generations, $g_{\max}$, is reached. Once the final set of solutions is generated, a set of Pareto Optimal solutions is obtained where each solution represent a optimized summary. Note that the sentences in the generated summary follow the same order as the original document.

5 Experimental Setup

This section includes the discussion about the dataset, the evaluation metrics used, the parameter settings and the comparative methods.

5.1 Dataset

A collection of original documents along with their reference summaries are necessary for assessing summarization algorithms. To evaluate, we have used a corpus of 50 Indian legal case documents from the Indian Supreme Court. This was provided by Bhattacharya et al. in their work [3]. The dataset was available with gold standard summaries which was annotated by two senior law students from the Rajiv Gandhi School of Intellectual Property Law. This institution is renowned for its excellence in legal education in India. These summaries are extractive in nature and approximately one-third the length of the documents and thus, served as the reference summaries for our obtained summaries evaluation purpose.

5.2 Metrics For Evaluation

To assess the performance and similarity of the generated summaries with the actual/gold summaries, we have used the ROUGE-N measure. This metric helps in measuring the overlap between the gold summary and the generated summary. A higher ROUGE score indicates a closer alignment between the two summaries. In our study, we considered values of N as 2, and L for ROUGE-2, and ROUGE-L, respectively, and uses ROUGE-2 Recall, ROUGE-2 F1, ROUGE-L Recall, and ROUGE-L F1-scores. For ROUGE mathematical definition, please refer to [22].

5.3 Parameter Settings

In our proposed framework, we utilized specific parameter values, such as a population size of 25 for the Differential Evolution (DE) algorithm, a mating pool size of 5, a mating pool construction threshold probability (β) of 0.8, a maximum of 25 generations ($g_{\max}$), a crossover probability (CR) of 0.8, $b = 6$, and $F = 0.8$. Additionally, we defined minimum ($S_{\min}$) and maximum ($S_{\max}$) thresholds for the number of words in a summary. The Word Mover Distance employs pre-trained InLegalBERT and InCaseBERT [19] models to compute sentence distances. We averaged results over 5 algorithm runs for each legal document. Generally, in evolutionary-based optimization algorithms, the number of fitness function evaluations (NFE) is used as a stopping criterion. In our case, NFE equaled 625. This is computed as the product of the population size and the maximum number of generations ($|P| \times g_{\max}$).

5.4 Comparative Methods

For the purpose of comparative study, we have only used extractive methods which belongs to different categories such as unsupervised, supervised, domain-specific, domain-independent and neural network-based methods. These include LexRank [7], LSA [6], Luhn [13], Reduction [9], LetSum [8], KMM [24], CaseSummarizer [20], MMR [26], DELSumm [2], SummaRuNNer [18], BERTSUM [12] and Gist [11]. Among these, DELSumm is the most recent. It is an extractive summarization technique which incorporates domain knowledge to create summaries from Indian legal case documents. Apart from these, we have also developed the single-objective version of our proposed algorithm.

Table 1. ROUGE scores for various methods

Type of Method	ROUGE-2		ROUGE-L	
	F1	Recall	F1	Recall
ϑ_1+ϑ_2+ϑ_3 *(proposed)*	**0.4892**	**0.4913**	**0.6794**	**0.6843**
LexRank [7]	0.3719	0.3662	0.5392	0.5068
LSA [6]	0.3646	0.3644	0.5483	0.5632
Luhn [13]	0.3873	0.3907	0.5521	0.5424
Reduction [9]	0.3814	0.3778	0.5420	0.5194
LetSum [8]	0.4137	0.4030	0.5846	0.5898
KMM [24]	0.3546	0.3540	0.5385	0.5407
CaseSummarizer [20]	0.3765	0.3699	0.5349	0.4925
MMR [26]	0.3729	0.3733	0.5680	0.6064
DELSumm [2]	0.4217	0.4323	0.6017	0.6831
SummaRuNNer [18]	0.4149	0.4104	0.5821	0.5835
BERTSUM [12]	0.4048	0.4044	0.5529	0.5600
Gist [11]	0.3593	0.3567	0.5501	0.5712

6 Experimental Results and Discussions

This section discusses the results obtained using our single and multi-objective optimizatioon framework. The relevance of using different objective functions is also provided. We will consider (ϑ_1) as the tf-idf scores, (ϑ_2) as the number of Indian Legal Catchphrases, and (ϑ_3) as the anti-redundancy measures. The results obtained using single objective function and multi-objective functions are illustrated in Fig. 1. The comparative results with existing methods are shown in Table 1.

6.1 Quantitative Analysis Using Different Objective Functions

Initially, we consider single objective functions, where the inclusion of (ϑ_2) outperforms (ϑ_1). This suggests that emphasizing legal terminology contributes more significantly to summary quality compared to solely relying on term frequency-inverse document frequency. Expanding to two-objective functions, combining ϑ_1 with ϑ_2 yields substantial improvements over individual methods. This indicates that integrating both leads to more informative and contextually relevant summaries. Similarly, when ϑ_1 is optimized along with ϑ_3, further enhancements are observed. This highlights the importance of reducing redundancy in legal summaries. Moreover, the three-objective approach (ϑ_1, ϑ_2, ϑ_3) achieves a harmonious balance between coherence, informativeness, and coverage. By incorporating diverse linguistic aspects, including legal terminology, term importance, and sentence uniqueness, our method ensures comprehensive and contextually accurate summaries. Overall, the qualitative analysis underscores the importance of considering multiple criteria in the summarization process to optimize summary quality and relevance. Through a systematic comparison of objective functions, our approach demonstrates its effectiveness in generating concise yet informative legal summaries. More details are provided below:

Using Single Objective Function. From Fig. 1, its clear that the results obtained by ϑ_1 and ϑ_2. Comparing the two objectives, ϑ_2 shows a 7.67% improvement in ROUGE-2 F1 score and 1.43% enhancement in ROUGE-L F1 score over ϑ_1. Similarly, based on ROUGE-2 Recall and ROUGE-L Recall scores, ϑ_2 outperforms ϑ_1 by 2.05% and 1.24%, respectively. This suggests that prioritizing the inclusion of legal catchphrases leads to better summarization results than relying solely on tf-idf scores. The higher performance of ϑ_2 may indicate that legal catchphrases contribute more significantly to the overall quality and relevance of the summary.

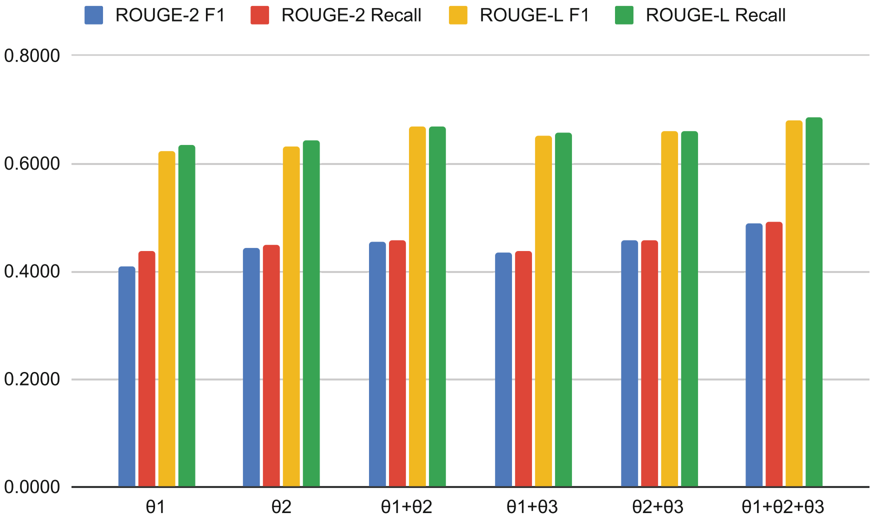


Fig. 1. Variation of different evaluation metrics using different combinations of objective functions ϑ_1, ϑ_2, and ϑ_3.

Using Multi-objective Functions. When employing multi-objective functions, the summarization process considers multiple criteria simultaneously to optimize the summary quality. This approach offers a more comprehensive assessment by balancing various aspects such as the tf-idf score, the number of legal catchphrases and anti-redundancy. From Fig. 1, we observe that the 2-objective functions combination outperforms individual (single objective) methods by significant margins. This is evidenced by the percentage increases in ROUGE-2 and ROUGE-L scores. Compared to ϑ_1 alone, (ϑ_1, ϑ_2) demonstrate a 11.06% increase in ROUGE-2 F1 score, 4.46% increase in ROUGE-2 recall, 7.23% increase in ROUGE-L F1 score and 5.41% increase in ROUGE-L recall. Similarly, when compared to ϑ_2, (ϑ_1, ϑ_2) exhibits improvements, with a 3.14% increase in ROUGE-2 F1 score and 2.37% increase in recall along with 5.72% increase in ROUGE-L F1 score and a 4.12% increase in ROUGE-L recall. In the same way, when compared to using ϑ_1, (ϑ_1, ϑ_3) achieves improvements, showing an improvement of 5.92% and 4.34% in terms of ROUGE-2 F1 score and ROUGE-L F1 score. Likewise, contrasted with ϑ_2, the combination of (ϑ_2, ϑ_3) demonstrates enhancements, showing a 3.25%, 2.05%, 4.17% and 2.58% surge in ROUGE-2 F1 score, ROUGE-2 recall, ROUGE-L F1 score and ROUGE-L recall, respectively.

When comparing the performance of different methods by taking two objective functions such as (ϑ_1, ϑ_2), (ϑ_2, ϑ_3), (ϑ_3, ϑ_1), (ϑ_1, ϑ_2) exhibits better performance. (ϑ_1, ϑ_2) exhibits superior performance over (ϑ_1, ϑ_3), showcasing significant percentage increases in ROUGE-2 F1 score (4.85%), ROUGE-2 Recall (4.96%), ROUGE-L F1 score (2.77%), and ROUGE-L Recall (1.81%). Additionally, while the percentage increases over (ϑ_3, ϑ_2) are relatively smaller, (ϑ_1, ϑ_2) maintains a slight edge, displaying improvements in ROUGE-2 Recall (0.31%), ROUGE-L F1 score (1.48%), and ROUGE-L Recall (1.50%). These findings underscore (ϑ_1, ϑ_2) efficacy in enhancing summarization quality, particularly in capturing relevance of the content and legal term coverage. This enables in creating complete, concise and informative legal summaries.

When comparing the three-objective method ($\vartheta_1, \vartheta_2, \vartheta_3$) to the two-objective methods (ϑ_1, ϑ_2), (ϑ_1, ϑ_3), and (ϑ_2, ϑ_3), the superiority of the three-objective approach in summarization quality becomes evident. With respect to Rouge-2 F1 scores, there is an enhancement of 10.69% over (ϑ_2, ϑ_3), 12.44% over (ϑ_1, ϑ_2), and 3.19% over (ϑ_1, ϑ_3). Similarly, Rouge-L F1 scores show an enhancement of 7.44%, 4.57%, and 2.93%, respectively. By integrating ϑ_1, ϑ_2, and ϑ_3, the three-objective approach achieves a harmonious balance between coherence, informativeness, and coverage by leveraging different linguistic and semantic aspects of the Indian Legal Case document. This helps in creating more comprehensive and informative summaries. It incorporates anti-redundancy to include legal sentences which are dissimilar or unique, it incorporates Indian Legal Case catchphrases to capture the essence of legal information contained in the Case documents and tf-idf scores to capture the important words in the document. Thus, the three-objective method emerges as the preferred choice.

The outcomes achieved through the simultaneous optimization of objective functions, denoted as ϑ_1, ϑ_2, and ϑ_3, demonstrate notable advancements compared to the DELSumm model for summarizing a dataset comprising 50 Indian Legal Case documents. For benchmarking, we have employed the DELSumm model, recognized for its efficacy in extractive Indian Legal Case summarization by leveraging domain expertise. The results reveal a clear superiority of our approach across all evaluation metrics. Specifically, our method showcases improvements of 16.00%, 12.91%, 13.65%, and 0.18% over the DELSumm model in terms of ROUGE-2 F1, ROUGE-L F1, ROUGE-2 Recall, and ROUGE-L Recall, respectively. This substantiates the efficacy of incorporating sentences

Table 2. Comparison of Actual/Reference Summaries with the Generated Summary

Actual/Reference Summary (Annotator 1)	Actual/Reference Summary (Annotator 2)	Generated Summary
this appeal is by special leave against the judgment of the high court of punjab and haryana confirming the conviction of the accused. the facts of the case in brief are that in view of the chinese invasion air field at sirsa required to be extended for which purpose the ministry of defence, govt. of india took steps to acquire some lands of agriculturists pursuant to which a notification dated november 27, 1962 was issued under section 4 of the land acquisition act 1894 for acquiring 51.79 acres of land situated in the state of ahmedpur	this appeal is by special leave against the judgment of the high court of punjab and haryana confirming the conviction of the accused. in view of the chinese invasion air field at sirsa required to be extended for which purpose the ministry of defence, govt. of india took steps to acquire some lands of agriculturists pursuant to which a notification dated november 27, 1962 was issued under section 4 of the land acquisition act 1894 for acquiring 51.79 acres of land situated in the state of ahmedpur	this appeal is by special leave against the judgment of the high court of punjab and haryana confirming the conviction of the accused under section 51c of the prevention of corruption act 1947. the facts of the case in brief are that in view of the chinese invasion air field at sirsa required to be extended for which purpose the ministry of defence govt. of india took steps to acquire some lands of agriculturists pursuant to which a notification dated november 27 1962 was issued under section 4 of the land acquisition act 1894 for acquiring 51 79 acres of land situated in the state of ahmedpur

enriched with maximum legal catchphrases, tf-idf scores, and anti-redundancy measures. This yields to more succinct yet comprehensive legal summaries. By integrating these diverse objective functions, our approach ensures that the generated summaries are not only contextually accurate but also comprehensive and informative. Thus, it caters to the nuanced requirements of legal practitioners and stakeholders. Furthermore, we have rigorously verified the statistical significance of the enhancements realized by our proposed methodology. It is quite notable that the best result obtained (reported in Table 1 is found to be statistically significant at 5% significance level.

Table 2 gives an example of the best summary generated by our method. We compare it with the reference/actual summaries provide by annotators 1 and 2 in the used dataset. The highest-scoring summary achieves average maximum scores of 0.8358 and 0.5488 for ROUGE-2 Recall and F1 scores, 0.9187 and 0.7217 for ROUGE-L Recall and F1 scores, respectively. It presents a clear and concise summary of the case with relevant details. As clearly visible, the information is in a structured manner and it stays focused on the main points of the case.

7 Conclusion and Future Works

This research presents an innovative unsupervised technique for generating extractive summaries of Indian legal case documents, utilizing a sophisticated

Multi- Objective Optimization (MOO) framework. Our approach redefines the summarization problem by framing it as a binary optimization task. The primary goal is to select the most pertinent sentences from a vast corpus of legal text while adhering to a defined word count limit, thus ensuring concise and relevant summaries. The core of our methodology lies in leveraging a multi-objective evolutionary algorithm, which allows for the concurrent optimization of several objective functions namely, tf-idf score, legal catchphrases, and anti-redundancy. For experimental purpose, we used legal case documents sourced from the Indian Supreme Court for evaluation. The results are validated using standard ROUGE scores, including ROUGE-2 F1, ROUGE-L F1, ROUGE-2 Recall, and ROUGE-L Recall. The results obtained reveals that our proposed algorithm significantly outperforms current state-of-the-art methods in generating extractive summaries. This is evident from the substantial improvements observed in various ROUGE scores, which are standard metrics for evaluating summarization quality. The ROUGE-2 F1 score, which measures the overlap of bigrams between the generated summary and the reference summary, showed an improvement of 16.00%. On the other hand, ROUGE-L F1 score, which assesses the longest common subsequence between the generated summary and the reference summary, showed an improvement of 12.91%. The overall improvements across multiple ROUGE metrics demonstrate the effectiveness of our multi-objective optimization framework in balancing various criteria such as term importance, presence of legal catchphrases, and anti-redundancy measures.

Despite the promising results, there are several avenues for future research that could further enhance the capabilities of our summarization system: (a) Expansion to Multi-Lingual and Cross-Jurisdictional Summarization: While our approach focuses on Indian legal documents, extending the methodology to encompass legal texts from multiple languages and jurisdictions could significantly broaden its applicability. Incorporating legal terminology and structures from different legal systems would enhance its versatility and utility; (b)User Feedback Incorporation: Incorporating user feedback into the optimization process could help tailor summaries to specific legal professionals' needs. Implementing interactive systems where users can refine or adjust summaries based on their preferences could further enhance the system's usability and accuracy.

References

1. Bhattacharya, P., Hiware, K., Rajgaria, S., Pochhi, N., Ghosh, K., Ghosh, S.: A comparative study of summarization algorithms applied to legal case judgments. In: Proceedings of the European Conference on Information Retrieval (2019)
2. Bhattacharya, P., Poddar, S., Rudra, K., Ghosh, K., Ghosh, S.: Incorporating domain knowledge for extractive summarization of legal case documents. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law, ICAIL 2021, pp. 22–31. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3462757.3466092
3. Cheng, J., Lapata, M.: Neural summarization by extracting sentences and words. In: Proceedings of the Annual Meeting of the Association for Computational Linguistics (2016)

4. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Parallel Problem Solving from Nature PPSN VI: 6th International Conference Paris, France, 18–20 September 2000 Proceedings 6, pp. 849–858. Springer (2000)
5. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the NAACL-HLT (2019)
6. Dong, Y.: A survey on neural network-based summarization methods. CoRR (2018). arXiv:1804.04589
7. Erkan, G., Radev, D.R.: Lexrank: graph-based lexical centrality as salience in text summarization. J. Artif. Int. Res. **22**(1), 457–479 (2004)
8. Farzindar, A., Lapalme, G.: Letsum, an automatic legal text summarizing system. In: Proceedings of the Legal Knowledge and Information Systems (JURIX) (2004)
9. Jing, H.: Sentence reduction for automatic text summarization. In: Proceedings of the Applied Natural Language Processing Conference (2000)
10. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: Proceedings of the International Conference on Machine Learning, pp. 957–966 (2015)
11. Liu, C.L., Chen, K.C.: Extracting the gist of Chinese judgments of the supreme court. In: Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL) (2019)
12. Liu, Y., Lapata, M.: Text summarization with pretrained encoders. In: Proceedings of the EMNLP-IJCNLP (2019)
13. Luhn, H.: The automatic creation of literature abstracts. IBM J. Res. Dev. **2**(2), 159–165 (1958)
14. Mallick, C., Das, A.K., Ding, W., Nayak, J.: Ensemble summarization of biomedical articles integrating clustering and multi-objective evolutionary algorithms. Appl. Soft Comput. **106**, 107347 (2021)
15. Mandal, A., Ghosh, K., Pal, A., Ghosh, S.: Automatic catchphrase identification from legal court case documents. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 2187–2190. CIKM 2017. Association for Computing Machinery, New York (2017). https://doi.org/10.1145/3132847.3133102
16. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013). https://arxiv.org/abs/1301.3781
17. Mishra, S.K., Harshavardhan, K., Mitra, S., Saha, S., Bhattacharyya, P.: Bug report summarization using multi-view multi-objective optimization framework. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1245–1253 (2022)
18. Nallapati, R., Zhai, F., Zhou, B.: Summarunner: a recurrent neural network based sequence model for extractive summarization of documents. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31 (2017)
19. Paul, S., Mandal, A., Goyal, P., Ghosh, S.: Pre-trained language models for the legal domain: a case study on Indian law. In: Proceedings of the 19th International Conference on Artificial Intelligence and Law (ICAIL 2023) (2023). https://arxiv.org/abs/2209.06049
20. Polsley, S., Jhunjhunwala, P., Huang, R.: Casesummarizer: a system for automated summarization of legal texts. In: Proceedings of the International Conference on Computational Linguistics (COLING) System Demonstrations (2016)

21. Saini, N., Kumar, S., Saha, S., Bhattacharyya, P.: Mining graph-based features in multi-objective framework for microblog summarization. In: 2020 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8. IEEE (2020)
22. Saini, N., Saha, S., Bhattacharyya, P.: Multiobjective-based approach for microblog summarization. IEEE Trans. Comput. Soc. Syst. **6**(6), 1219–1231 (2019). https://doi.org/10.1109/TCSS.2019.2945172
23. Saini, N., Saha, S., Bhattacharyya, P., Tuteja, H.: Textual entailment–based figure summarization for biomedical articles. ACM Trans. Multimed. Comput. Commun. Appl. (TOMM) **16**(1s), 1–24 (2020)
24. Saravanan, M., Ravindran, B., Raman, S.: Improving legal document summarization using graphical models. In: Legal Knowledge and Information Systems. JURIX (2006)
25. Wu, H.C., Luk, R.W.P., Wong, K.F., Kwok, K.L.: Interpreting TF-IDF term weights as making relevance decisions. ACM Trans. Inf. Syst. (TOIS) **26**(3), 1–37 (2008)
26. Zhong, L., Zhong, Z., Zhao, Z., Wang, S., Ashley, K.D., Grabmair, M.: Automatic summarization of legal decisions using iterative masking of predictive sentences. In: Proceedings of the International Conference on Artificial Intelligence and Law (ICAIL) (2019)
27. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthan, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol. Comput. **1**(1), 32–49 (2011)

VisEmoComic: Visual Emotion Recognition in Comics Image

Ruddy Théodose(✉) and Jean-Christophe Burie

L3i Laboratory, SAIL Joint Laboratory, La Rochelle Université,
17042 La Rochelle CEDEX 1, France
{ruddy.theodose,jean-christophe.burie}@univ-lr.fr

Abstract. Emotion recognition in images have bean widely studied on captured data of real people but few works have been realized on drawn data. Among this category, comic books have become an important part of the of the popular culture. Whether realistic drawings or oversimplified designs, characters have to depict credible and understandable reactions to the events of the story they are included in. While human-like characters designs are often inspired by real face mechanisms, authors may include various graphic elements to emphasize those reactions to the events they undergo. In this paper, we propose VisEmoComic, an image-based dataset for emotion recognition on comics. Several annotators were invited to give their interpretation of the character emotions represented in given scenes. The image data comes from existing comic book datasets, dedicated to other tasks and from various origins, allowing to include cultural specificities. Additionally, for each sample, the face of the character of interest, its body and the frame where it was drawn are given to allow the use of the immediate spatial context for prediction. Collected samples were annotated by multiple annotators. Consequently, we proposed two schemes to generate labels that sum up the man-made labels and defined baselines using the built dataset.

Keywords: Emotion Recognition · Manga · Comics Analysis · Document Analysis

1 Introduction

Non verbal communication is a key element to understand all the nuances a discourse can bring. When it comes to direct communication between persons, cues such as posture, gestures, voice intonation, gaze, or facial expressions allow to interpret details in the told speeches. Consequently the field of affective computing has explored the exploitation of these hints for various application topics such as behavioural analysis in crowds, human-machine interfaces or customers/testers satisfaction evaluation.

One subtopic of affective computing concerns the perception of the human agent's emotional state. While studies on captured data (acquired through sensors) have been widely investigated, with numerous methods and datasets [1,18],

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 281–296, 2025.
https://doi.org/10.1007/978-3-031-78495-8_18

pictorial images have drawn less attention. Among the various forms of visual arts, comics, also known as "sequential art" is a major part of the popular culture. They illustrate characters acting in a story told through successions of static panels. The design of drawn characters is often inspired by how real bodies look like and work, so we might suppose that techniques developed for captured data may also be applied to drawn characters. Moreover, besides the body related cues, comic artists have produced visual tools specific to the medium such as symbols or background effects in order to accentuate various aspects of the narration.

Comics analysis community have grown over the last decade. Detection of structural elements inside pages such as panels [16], characters [8,27], speech bubble and text lines [9] and even onomatopoeia [24] have been heavily investigated. Understanding what is described inside panels depends on the comprehension of social and cultural factors that may greatly vary between authors and between readers. In that direction, understanding the character's behaviour is a crucial topic to extract insights for higher goals such as the development of more efficient translation tools or for accessibility purposes.

In this paper, we introduce VisEmoComic, a new dataset for visual emotion recognition in comics.

Building on the foundation established by the Kangaiset dataset, this work continues the development of resources specifically for emotion analysis in comic media.

The interpretation of emotions remains a highly subjective topic, dependant on numerous components such as reader's cultural background. Consequently, the dataset was annotated by multiple annotators in order to gather possibly different opinions on the same data.

Moreover, we trained different networks on the built in order to create baselines. Multiple annotators were involved. One approach would be to train one model on each annotator's label. However, this approach is hardly scalable and can potentially overfit the biases of this annotator. Instead, we proposed to generate intermediate labels that attempt to summarize the "human" labels. The networks trained on generated labels are also evaluated on the original ones to analyze the proximity of the produced annotations with the man-made references. EmoRecCom challenge [29] have realized similar task by estimating the displayed emotions in the whole panel through the combination of text in bubbles and image. Most of the proposed methods combined image processing and language models. In this paper, we restrict to visual modality as we aim at studying the influence of graphical cues on methods initially developed for processing photos. Our contributions are:

- We have built an image-based dataset for emotion recognition on comics from various countries that provides boxes for face and body of the characters of interest;
- We trained and evaluated multiple models, initially designed for processing real captured data on this new dataset using two training strategies that consider the varying opinions of different annotators.

2 Related Work

In this section, we first review existing datasets used in general comic analysis. Then we introduce several existing methods for emotion recognition based on real captured data, with a particular emphasis on methods that prioritize image-based analysis.

2.1 Emotion Recognition

Emotion Recognition has been heavily investigated on various types of data.

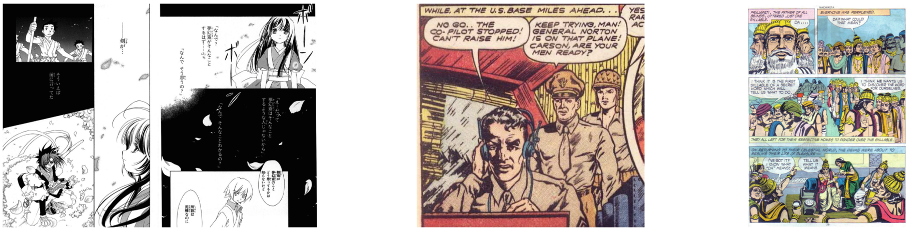
About image data, face images were considered to provide the most critical information for emotion recognition. When only face data is used, the acronym FER for Facial Expression/Emotion Recognition is often employed in the literature. Prior works used handcrafted features such as Local Binary Patterns [33] before exploring deep learning techniques [22]. While the task is related to object classification, specific techniques have been developed to better address its unique challenges. These techniques include new architectures such as Res-Masking [30] that integrates U-Nets into ResNet blocks to generate attention maps, as well as various loss functions [6,11,12], and ensemble networks [5] etc.

However, various works on psychology research [4] suggested the importance of contextual information for the realization of the task.

Following that assumption, research on (spatial) context aware emotion recognition (often abbreviated *CAER*) started to develop. Those methods tend to employ simultaneously multiple images. EMOTIC [19,20] is currently one of the largest public dataset on CAER. The authors of the dataset proposed a two-branch network, one branch focusing on extracting features related to the character of interest while the second branch deals with the whole image. A final fusion subnetwork is in charge of merging the extracted features to generate a prediction. CAER-Net [21] adopts a similar two-branch while estimating and applying weights to each branch before merging the weighted features. Mittal et al. [26] extends the multiple branch approach with EmotiCon by adding modalities other than image data such as character pose estimated by systems such as OpenPose [7] and depth image to better deal with spatial relationships in the scene. Context-Dependent Net (CD-Net) [35] computes global shared features from the entire image and use a transformer to aggregate face, body and scene information.

2.2 General Comics Databases

The most well-known public datasets for comics analysis and understanding mainly focus on detecting structural elements of pages, such as panels, characters, text, and speech bubbles. eBDThèque [13] was one of the earliest published dataset on the topic. It contains 100 pages from American, European and Japanese comics. By the same team, a dataset named DCM77 [28] focuses on American Golden Age comics available in the public domain in the Digital Comic Museum. Manga109 [2] is currently, to our knowledge, the largest dataset on the task. It focuses on the japanese untranslated mangas. Multiple extensions were developed around this dataset in order to study other comic related tasks. For example, COO dataset [3] focused on the detection of onomatopoeia and Manga109Dialog [23] provides label data for comic speaker detection. IMCDB (Indian Mythological Comic Database) [14] is a collection of Indian comics translated in English that contains data for panel, speech bubbles and transcriptions of text lines. However, none of these datasets are specifically designed for the task of emotion recognition.


WHILE, AT THE U.S. BASE MILES AHEAD. . .
NO GO.. THE CO-PILOT STOPPED! CAN'T RAISE HIM!
KEEP TRYING, MAN! GENERAL NORTON IS ON THAT PLANE! CARSON, ARE YOUR MEN READY?

(a) Manga109 (b) EmoRecCom (c) IMCDB

Fig. 1. Unprocessed data from the studied datasets, Manga109 are double pages, EmoRecCom images are panels, IMCDB images are single pages.

3 Dataset

In this section, we describe the construction process of the VisEmoComic dataset and provide statistical information on the aggregated data.

3.1 Data Construction

Data Sources. The dataset was created under the assumption that facial expressions of emotions are universal across cultures, even though psychologists are still debating about this topic [17]. Moreover, drawn representations may also greatly vary according the cultural background of the artists. For this reason, we extracted images from the Japanese mangas in Manga109 dataset [2,25], American Golden Age comics from the EmoRecCom Challenge dataset [29] and Indian comics from IMCDB dataset [14] into a single dataset for emotion recognition Fig. 1. The Manga109 and IMCDB datasets do not include emotion labels, as they were not created for the purpose of emotion recognition. The EmoRecCom challenge was created specifically for emotion recognition on comics. However, the provided labels consider the overall mood of the scene rather than emotions assigned to individual characters. Consequently, there are no specific emotion labels assigned to each character separately.

Dataset Structure. In this paper, we call "character image" the image related to the character of interest, either the face image or the body image, in opposition to panel image. as illustrated in Fig. 2. Each entry of the dataset represents a specific character of interest in a given situation represented by the panel. One sample then contains the panel, the location of the face and the body of the character of interest inside this panel. Manga109 dataset already provides the whole pages and for each panel, face and body boxes are given with the identifier of the represented character. For EmoRecCom, we used some of the panels extracted by the authors of the challenge. However, as aforementioned, neither face nor body boxes are available so we used a YOLOv5 network trained on Manga109 boxes to produce characters' boxes. The associations between face and body boxes were done manually. In IMCDB dataset, pages are available, but not all of them have panel boxes. Thus, the available panels were extracted and processed with the same approach as the one used for EmoRecCom panels. We built this dataset with the assumption that faces are required for the emotion estimation on a visual standpoint. Characters seen from the back were not integrated.

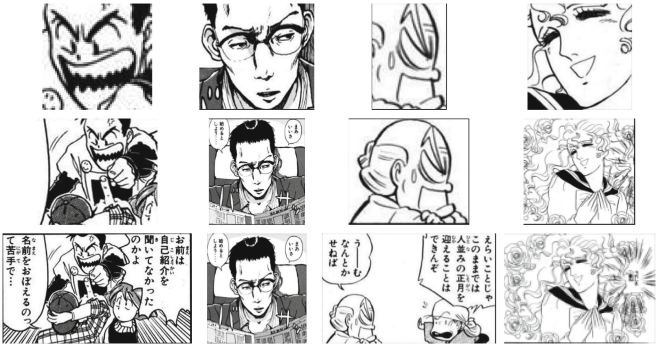

お前は自己紹介を聞いてなかったのかよ
名前をおぼえるのって苦手で…
まあいいさ
始めるとしよう……
えらいことじゃこのままでは人並みの正月を迎えることはできんぞ
う——むなんとかせねば

Fig. 2. Illustration of the different scales of study, rows: face, body and panel. Data extracted from Manga109 dataset. Columns in order: Appare Kappore ©Kanno Hiroshi, Hanzai Kousyounin Minegishi Eitarou ©Ki Takashi, Jiji Baba Fight ©Nishikawa Shinji, Karappo Highschool ©Takaguchi Satosumi

Emotion Model. Literature has modeled the concept of emotion in various ways. Methods such as Plutchik model [31] define a discrete and finite set while others such as the Russell's circumplex model [32] tend to represent emotions as points in multidimensional space. For the dataset, we use the Ekman model [10] that divide the emotion spectrum into 6 classes besides the neutral expression: anger, disgust, fear, joy, sadness and surprise. The inclusion of a "neutral" class is subject to debate since it could be perceived as the absence of a strong reaction, and therefore, not strictly classified as an emotion. While more complex models such as the aforementioned ones exist, we wanted a set that are simple enough to be accessible to non expert annotators.

Annotation. For each dataset, 3 annotators were chosen. The three datasets were not necessarily processed by the same group because the annotators had to master the language of their assigned sub-dataset. Even if we didn't planned, at this stage, to use textual information, we considered that reading what was said by the characters could provide additional information on the events and help the annotators in their decision. Annotators received the panels showing the character of interest framed in a colored rectangle and were asked to choose between the given emotions. Multilabel classification was authorized meaning that annotators could select multiple choices. This decision was made in order to mitigate the smaller representation capability of a single label Ekman-based classification. Moreover, situations that the characters undergo induce complex reactions, so multilabel classification reduces the number of constraints imposed on the annotators who do not have to select a single most "predominant" emotion. Annotators were asked to select one or two classes, three in the most extreme cases.

3.2 Data Statistics

Table 1 shows the composition of each sub-datasets per annotator. The first observation is the large class imbalance. Data is extracted directly from comic books meaning that characters are integrated inside a story and their actions and reactions have to be coherent with the events. For all annotators and all subsets, the "disgust" class is poorly represented while "neutral" and "joy" are the most dominant emotions.

Table 2 compares the Kangaiset dataset [34] with our VisEmoComic dataset, highlighting two main differences.

Firstly, Kangaiset confines its data to Manga109 pages, whereas we expanded our dataset to include images from the EmoRecCom and IMCDB datasets, resulting in greater diversity in graphic style representations.

Secondly, there is a distinction in the annotation process. The creators of Kangaiset used a single annotator for label creation, whereas our annotation process involved three specialized annotators.

3.3 Agreement Between Annotators

As the topic remains subjective, it is interesting to measure how similar are the labels produced by the different annotators. Several metrics exists to measure this similarity. Here, we use the Cohen's kappa and the Fleiss' kappa, two chance-corrected coefficients. Both produce a score between -1 and 1, -1 meaning a complete disagreement between annotators, 0 means that the labels were produced randomly and 1 a complete agreement. However, the interpretation of the intermediate values may vary between specialists. Cohen's kappa can be computed only between two annotators while Fleiss' kappa allows to compute a score between two or more annotators. The computed scores are displayed in Table 3. For all classes, Manga109 images seem to be more consensual than EmoRecCom and IMCDB data. One assumption could be the design and scenography

Table 1. Emotion count for the three datasets. Due to the multilabel setup, the sum of each row may not match the number of samples. Manga109: 12616 samples; EmoRecCom: 3609 samples; IMCDB: 1036 samples

Dataset	AnnID	Anger	Disgust	Fear	Joy	Neutral	Sadness	Surprise
Manga109	1	2335	186	1483	3175	1625	1451	3113
	2	2616	60	1185	3117	1116	1924	3364
	3	2597	150	1397	3404	1758	1662	3258
EmoRecCom	1	817	174	787	849	311	190	783
	2	988	225	597	709	488	175	785
	3	815	176	639	970	628	272	425
IMCDB	1	164	11	61	324	231	45	267
	2	185	17	42	228	245	89	293
	3	155	7	67	303	325	91	123

Table 2. Comparison between Kangaiset and VisEmoComic. The three last columns represent the number of samples extracted from each datasets.

Dataset	Annotator Number	Manga109	EmoRecCom	IMCDB
Kangaiset	1	9387	/	/
VisEmoComic	3	12616	3609	1036

trends. As illustrated in Fig. 3 that display the histogram of the ratios face box size/panel size for the three subsets, Japanese mangas tend to prefer close shots with more refined faces meaning that characters are more easily recognizable. EmoRecCom and IMCDB images often include large shots meaning that actions and scenes tend to predominates over the characters themselves, at least on the visual aspect and for the selected data. Moreover, we can observe that the scores for the “disgust” class are consistently close to 0. This indicates a scarcity of annotations across all raters, making it challenging to discern a definitive trend. Consequently, the metrics suggest that these annotations are more likely to have been assigned randomly.

4 Experiments

4.1 Tested Networks

In this paper, multiple methods were evaluated. First, we assessed FER methods that use only face images. We evaluated a Resnet34 [15], a Resnet34 with spatial and channel attention (CBAM) [36] and a ResnetMasking34 [30]. Then, we evaluated three CAER methods, which simultaneously use the image of a character (face or body) as well as data from its spatial context. This category includes the CAER-S network [21], the EMOTIC network [20], and the CD-Net [35].

Table 3. Agreement scores on the different datasets. “Cohen 1v2” is the Cohen’s kappa computed with the labels of annotator n°1 and n°2

Dataset	Metric	Anger	Disgust	Fear	Joy	Neutral	Sadness	Surprise
Manga109	Cohen 1v2	0.7597	0.2876	0.4836	0.8916	0.517	0.6511	0.6049
	Cohen 1v3	0.7456	0.1073	0.4811	0.8892	0.5662	0.6971	0.5656
	Cohen 2v3	0.7043	0.08905	0.4232	0.8634	0.4397	0.6233	0.541
	Fleiss	0.736	0.1579	0.4631	0.8813	0.5091	0.6553	0.5703
EmoRecCom	Cohen 1v2	0.6575	0.2393	0.5354	0.773	0.4699	0.5528	0.5666
	Cohen 1v3	0.5091	0.1532	0.4823	0.6969	0.2742	0.3954	0.3591
	Cohen 2v3	0.5172	0.227	0.5686	0.7126	0.3912	0.3462	0.4459
	Fleiss	0.5619	0.2084	0.5264	0.7255	0.3712	0.4229	0.4601
IMCDB	Cohen 1v2	0.7555	0.204	0.3371	0.7118	0.6346	0.4931	0.6137
	Cohen 1v3	0.6999	0.1037	0.5005	0.7783	0.5475	0.4067	0.4672
	Cohen 2v3	0.6838	-0.009665	0.411	0.7158	0.5532	0.4524	0.4169
	Fleiss	0.7135	0.1042	0.4213	0.7354	0.5745	0.449	0.499

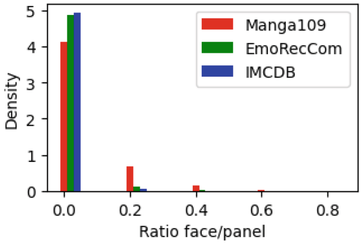


Fig. 3. Histograms of ratios face/panels for each dataset, x-axes represent the computed ratios, y-axes represent densities.

4.2 Training and Testing Setups

Manga109 and IMCDB provides information on the books where pages come from. EmoRecCom panels are not linked to any book information but the file naming convention provides hints on the original sources. The three subsets are split into a train and a test set. However, instead of splitting the whole batch of images, we split the books. As authors have their own specific art styles, splitting the books allows to reproduce the cases when new and unknown books are processed. In this experiment, we opted for a 7:3 ratio for train:test set.

Each sample have been annotated by multiple annotators. However, for the training, a label has to be defined. We planned two different training schemes. The first one, named *Perm* for “Permissive” consider that if a class is selected by at least one annotator, it is included in the training label. For the second one named *Maj* for “Majority”, an emotion is included in the training label if at least two out of three annotators have selected the same emotion. Figure 4 provides examples of those generated labels according to the annotators' labels.

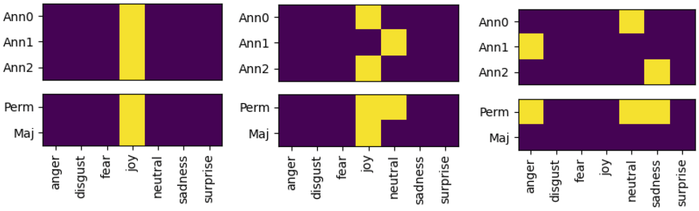


Fig. 4. (top): Human annotations; (bottom): generated labels under the two schemes. Left: Unanimity case; Middle: Slight differences between annotators; Right: Complete disagreement

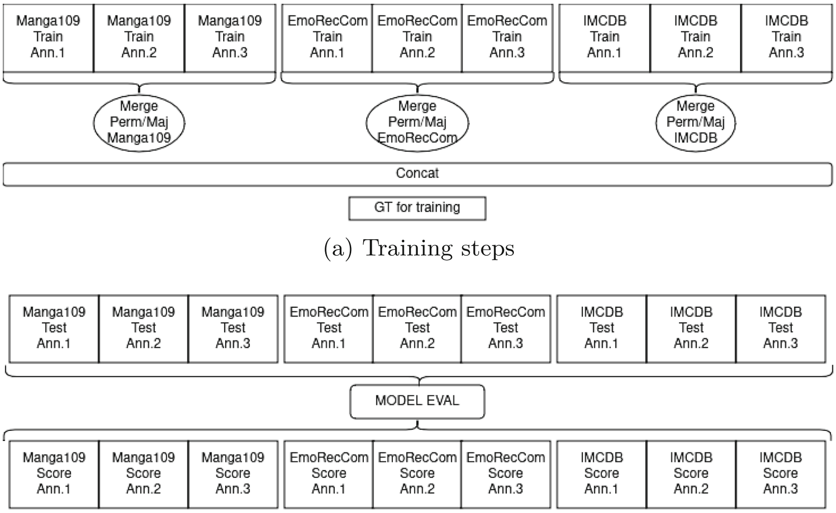
Figure 5 illustrates our training and testing processes. For the training phase (Fig. 5a), the annotation answers from each subset (Manga109, EmoRecCom, IMCDB) are merged into either majority or permissive labels then concatenated to create a complete training set. For the evaluation phase (Fig. 5b), the trained network is evaluated separately on each test subset using manual labels. We define $A = \{1, 2, 3\}$ the set of annotators, $D = \{$Manga109, EmoRecCom, IMCDB$\}$ the set of studied subsets and E the emotion classes (in our case, 6 emotions of the Ekman model + neutral). We note $S_{a,d,e}$ the test score for the annotator $a \in A$, on the dataset $d \in D$, and the emotion $e \in E$.

The metric used is the F1 score. Since the networks were trained for multilabel classification, we computed macro F1 scores (unweighted mean between F1 scores for positive and negative samples) for each emotion, consequently each category is processed independently as a binary classification.

Images related to characters are resized to 256 × 256 pixels. However, for panel images, since a scene can include multiple characters, maintaining the same size could compromise the visibility of smaller characters within the panel. Therefore, panel images are resized to 512 × 512 pixels. All networks were trained with the same hyperparameters: training lasted for 80 epochs using the focal loss function and the Adam optimizer, coupled with a one-cycle learning scheduler and a maximum learning rate set to 1e-4.

4.3 Global Results

In this section, we first evaluate global mean on the tested networks. The average score for one emotion is computed with $S_e = \frac{1}{\#D\#A} \sum_{d \in D} \sum_{a \in A} S_{a,d,e}$ and the average on all emotions $S = \frac{1}{\#E} \sum_{e \in E} S_e$. Table 4 summarizes the initial input configuration of all the tested networks. For ResMasking and the three CAER methods, the inputs setups match the one introduced in the original papers. All the macro F1 scores are listed in Table 5.



(a) Training steps

(b) Testing steps, bottom blocks represent $S_{a,d,e}$

Fig. 5. Our proposed training and testing processes.

Table 4. Base Input Configuration for each experiment

Model	Face	Body	Panel
Resnet34	✓		
Resnet34CBAM	✓		
ResMasking34	✓		
CAER-S Net	✓		✓
EMOTIC Net		✓	✓
CD-Net	✓	✓	✓

As expected, minor classes such as “disgust” and “fear” exhibit lower detection rates while major emotions such as “joy” and “anger” illustrate good performances for the all the tested networks. However, the “neutral” class yields comparatively lower results. Given that emotions are typically associated with facial expressions, the “neutral” class represents a challenge as it corresponds to the default facial expression without any emotional marker. Depending on one’s perspective on the “neutral” class, finding positive examples for “neutral” emotions and negative examples for the other six basic emotions can be considered analogous tasks. Consequently, trying to separate the “neutral” class could potentially complicate the training process. Interestingly, similar dynamics can be observed between agreement scores and F1 scores for each emotion. Kappa scores indicate the level of consensus among classes, with low agreement scores suggesting greater classification difficulty. For the “disgust” class, the scores also

Table 5. Global Macro F1 scores for each emotion S_e and average S.

Method	Anger	Disgust	Fear	Joy	Neutral	Sadness	Surprise	Mean
Resnet Perm	68.89	52.47	**61.94**	**79.25**	63.97	**62.55**	**64.67**	**64.82**
Resnet Maj	69.32	49.25	59.23	77.01	61.32	59.71	64.47	62.90
CBAM Perm	69.14	52.20	60.48	78.41	64.14	60.12	63.58	64.01
CBAM Maj	66.87	49.32	60.15	76.81	62.48	59.10	62.64	62.48
ResMasking Perm	69.05	49.32	60.40	77.19	62.45	60.65	64.44	63.36
ResMasking Maj	66.02	49.32	56.86	75.38	60.74	58.91	61.89	61.30
CAER-S Perm	**70.34**	50.31	61.30	78.52	**64.40**	61.49	64.30	64.38
CAER-S Maj	65.54	49.32	58.92	74.96	60.76	59.40	63.79	61.81
EMOTIC Perm	64.44	50.95	59.36	70.10	61.24	60.01	61.47	61.08
EMOTIC Maj	62.29	49.32	56.20	68.71	57.35	54.22	60.05	58.31
CDNet Perm	64.69	**53.81**	58.77	72.94	61.90	57.63	62.87	61.80
CDNet Maj	60.31	49.47	54.85	69.46	59.23	52.51	60.90	58.11

Table 6. F1 scores for **positive** and **negative** samples for the ResNet34 trained on permissive labels.

Metric	Anger	Disgust	Fear	Joy	Neutral	Sadness	Surprise	Mean
Neg F1	85.60	97.73	88.80	89.02	84.67	93.54	82.60	88.85
Pos F1	52.17	7.21	35.08	69.48	43.27	31.55	46.75	40.79
Macro F1	68.89	52.47	61.94	79.25	63.97	62.55	64.67	64.82

depends on the quantity of positive samples, but for the "neutral" class which is one of the major class, the F1 scores show that it was not the easiest "emotion" to classify.

Although the ResNet34 network is considerably simpler compared to the others, it demonstrates comparable performance on the task. This suggests that the image of the character alone conveys the most crucial emotional information. The addition of contextual information appears to have a minor impact on performance. However, these results should be interpreted with caution due to the chosen training conditions. The same hyperparameters were applied to all networks, without taking into account the conditions presented in the original papers or the inherent complexity of each network. While all experiments converged, the simplicity of the ResNet34 compared to the others may have contributed to its generally better performance.

While we displayed the macro F1, it is interesting to also analyze the gap between positive and negative F1 scores, displayed in Table 6. Negative samples are much more abundant, making it easier to sets predictions to zero. Figure 6 illustrates some predictions on the test sets. Even if they were trained under the same conditions, network predictions can vary significantly when major cues are not visible.

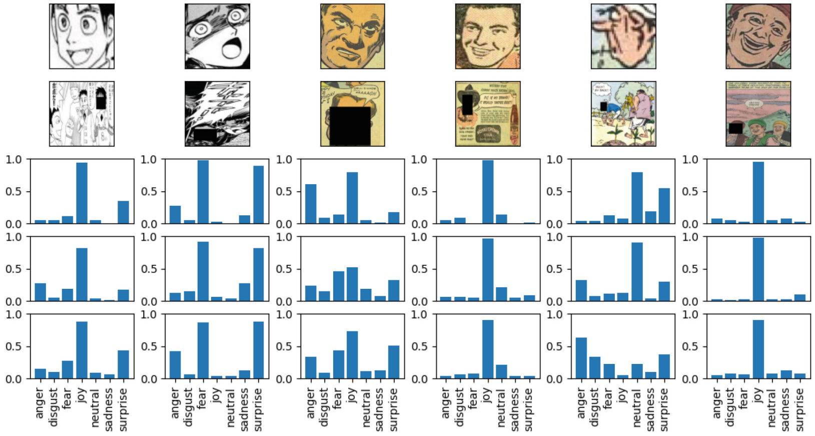


Fig. 6. Predictions from the three networks. Column 1–2: Manga109 data; Column 3–4: EmoRecCom data; Column 5–6: IMCDB data. Row 1: Face Image; Row 2: Panel image, the face is masked; Row 3–5: outputs from ResNet, CAER-S Net and EMOTIC Net in that order.

4.4 Results on Individual Annotators' Labels

In this section, we conduct a detailed analysis to examine the impact of the generated labels in relation to the individual annotators' labels across the different datasets. Given that ResNet34 produced the best overall performance in the previous section, our focus remains on this network. Table 7 presents the F1 scores for each dataset, annotator and emotion. ($S_{a,d,e}$ defined earlier).

We first observe that, for the same experiment, results tend to be better on Manga109 compared to EmoRecCom and IMCDB. This is likely due to the larger size of the Manga109 dataset used for training, which allows the network to specialize more effectively in this type of data. Additionally, faces in the EmoRecCom and IMCDB datasets are often less prominent, making it more challenging to analyze facial expressions when characters are not depicted in close-up shots.

For the three datasets, networks trained on "Permissive" labels demonstrate better alignment with individual opinions. In fact, a label set to "positive" by the "Majority" scheme indicates the number of annotators who have chosen this emotion, suggesting a higher level of consensus and thus easier fitting. However, this approach can also ignore the opinions expressed by the minority. If we consider the annotated emotions as sets, the "permissive" set encompasses the "majority" set for each image. One can suppose that permissive labels, by adapting to marginal judgments, may increase the risk of classification error for certain annotators. However, under the given training and testing conditions, the inclusion of all opinions seems to overtake the effects of potential noise.

Table 7. Macro F1 scores $S_{a,d,e}$ of ResNet34 on each subset and each annoator separately. "Dset": Dataset; "AnnID": Annotator ID; "M109": Manga109; "ERC": EmoRecCom

Dset	AnnID	Label	Anger	Disgust	Fear	Joy	Neutral	Sadness	Surprise	Mean
M109	1	Perm	78.02	53.41	65.70	89.27	73.96	73.12	75.07	72.65
		Maj	78.29	49.58	60.98	89.38	68.13	74.82	73.88	70.72
	2	Perm	78.20	49.78	63.65	88.50	66.23	72.04	74.45	70.41
		Maj	77.00	49.88	59.00	88.32	65.29	70.20	73.23	68.99
	3	Perm	77.70	49.60	63.58	88.98	71.45	74.76	72.76	71.26
		Maj	76.99	49.71	60.20	88.08	65.47	72.30	70.76	69.07
ERC	1	Perm	60.81	54.03	62.63	71.24	54.69	54.89	58.67	59.57
		Maj	62.73	48.47	59.06	71.39	59.25	49.58	56.90	58.20
	2	Perm	63.21	51.90	60.56	71.43	61.17	52.41	60.30	60.14
		Maj	62.83	48.10	62.78	72.65	55.43	50.13	55.47	58.20
	3	Perm	63.20	53.53	63.59	75.15	65.32	54.68	54.21	61.38
		Maj	64.20	48.64	63.94	71.91	55.66	49.09	56.84	58.61
IMCDB	1	Perm	65.39	49.18	60.34	74.82	59.21	60.97	65.83	62.25
		Maj	67.10	49.76	57.46	69.04	59.06	59.06	67.56	61.29
	2	Perm	66.67	61.63	57.51	75.90	61.53	58.62	63.41	63.61
		Maj	65.94	49.43	55.68	73.40	63.39	56.00	63.75	61.08
	3	Perm	66.80	49.13	59.92	77.97	62.21	61.43	57.37	62.12
		Maj	68.80	49.71	53.99	68.89	60.25	56.18	61.85	59.95

4.5 Difference Between Face and Body Images

Historically, facial features were considered as the main features for emotion recognition, leading to the development of the "FER" terminology for methods centered on this modality. However more recent context-aware and multimodal methods integrate bodily features such as pose or gait. In this section, we compare the impact of face and body images on prediction results. All the tested networks, except CD-Net which already use both, were trained on either face or body images, with permissive labels. Table 8 shows the global F1-scores S_e for each experiment.

In most cases, using body images instead of face images results in poorer performance.

While character poses may convey emotional cues, they are often more indicative of actions rather than emotions relevant to the task. When body images lack close-ups, crucial facial information becomes diluted in the "body" context, making the input data less relevant for the emotion recognition.

Table 8. F1 scores comparison between face and body inputs.

Method	Anger	Disgust	Fear	Joy	Neutral	Sadness	Surprise	Mean
Resnet Face	68.89	52.47	61.94	79.25	63.97	62.55	64.67	64.82
Resnet Body	65.57	50.65	60.91	71.85	61.16	59.63	62.92	61.81
CBAM Face	69.14	52.20	60.48	78.41	64.14	60.12	63.58	64.01
CBAM Body	60.99	50.19	56.89	71.25	59.73	59.01	62.30	60.05
ResMasking Face	69.05	49.32	60.40	77.19	62.45	60.65	64.44	63.36
ResMasking Body	56.98	49.32	55.36	69.75	57.92	52.74	59.05	57.30
CAER-S Face	70.34	50.31	61.30	78.52	64.40	61.49	64.30	64.38
CAER-S Body	64.88	49.78	59.21	71.52	60.93	60.15	61.46	61.13
EMOTIC Face	70.62	52.91	60.71	79.09	63.34	61.02	63.61	64.47
EMOTIC Body	64.44	50.95	59.36	70.10	61.24	60.01	61.47	61.08

5 Conclusion

In this paper, we introduce VisEmoComic, a novel dataset designed for visual emotion recognition in comics. The data collection process was curated from various sources to examine how emotions are represented across different cultures. Each character is analyzed at three levels, face, body and panel, in order to evaluate its emotional state. Given the subjectivity inherent in emotion recognition, we requested the opinion of multiple annotators to build the dataset and proposed schemes to train a system based on the "median" annotator's perspective rather than fitting it to a specific annotator's style. We established initial baselines using various networks, some of which incorporate the spatial context of the character of interest. While annotators were strongly encouraged to consider all panel elements, including text in speech bubbles, for labeling decisions, our primary aim is to explore what can be inferred solely from the image for emotion recognition. Future research may explore approaches that combine text and image processing to harness insights from both modalities.

Acknowledgements. This work benefited from access to the computing resources of the L3i laboratory, operated and hosted by the University of La Rochelle and the computing resources of the "CALI 3" cluster, operated and hosted by the University of Limoges, part of the HPC network in the Nouvelle-Aquitaine Region. Both are financed by the State and the Region.

References

1. Ahmed, N., Aghbari, Z.A., Girija, S.: A systematic survey on multimodal emotion recognition using learning algorithms. Intell. Syst. Appl. **17**, 200171 (2023)
2. Aizawa, K., et al.: Building a manga dataset "manga109" with annotations for multimedia applications. IEEE Multimed. **27**(2), 8–18 (2020)

3. Baek, J., Matsui, Y., Aizawa, K.: COO: comic onomatopoeia dataset for recognizing arbitrary or truncated texts. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) ECCV 2022. LNCS, vol. 13688, pp. 267–283. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19815-1_16
4. Barrett, L.F., Mesquita, B., Gendron, M.: Context in emotion perception. Curr. Dir. Psychol. Sci. **20**(5), 286–290 (2011)
5. Białek, C., Matiolański, A., Grega, M.: An efficient approach to face emotion recognition with convolutional neural networks. Electronics **12**(12), 2707 (2023)
6. Cai, J., Meng, Z., Khan, A.S., Li, Z., O'Reilly, J., Tong, Y.: Island loss for learning discriminative features in facial expression recognition. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), pp. 302–309. IEEE (2018)
7. Cao, Z., Simon, T., Wei, S.-E., Sheikh, Y.: Realtime multi-person 2D pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7291–7299 (2017)
8. Chu, W.-T., Li, W.-W.: Manga FaceNet: face detection in manga based on deep neural network. In: Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval, Bucharest Romania, pp. 412–415. ACM (2017)
9. Dubray, D., Laubrock, J.: Deep CNN-based speech balloon detection and segmentation for comic books. In: 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 1237–1243. IEEE (2019)
10. Ekman, P., Friesen, W.V.: Constants across cultures in the face and emotion. J. Pers. Soc. Psychol. **17**(2), 124–129 (1971)
11. Fard, A.P., Mahoor, M.H.: Ad-Corre: adaptive correlation-based loss for facial expression recognition in the wild. IEEE Access **10**, 26756–26768 (2022)
12. Farzaneh, A.H., Qi, X.: Facial expression recognition in the wild via deep attentive center loss. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2402–2411 (2021)
13. Guérin, C., et al.: eBDtheque: a representative database of comics. In: 2013 12th International Conference on Document Analysis and Recognition, pp. 1145–1149. IEEE (2013)
14. Gupta, V., Detani, V., Khokar, V., Chattopadhyay, C.: C2VNet: a deep learning framework towards comic strip to audio-visual scene synthesis. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12822, pp. 160–175. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86331-9_11
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
16. He, Z., et al.: An end-to-end quadrilateral regression network for comic panel extraction. In: Proceedings of the 26th ACM International Conference on Multimedia, MM 2018, pp. 887–895. Association for Computing Machinery, New York (2018)
17. Jack, R.E., Garrod, O.G.B., Yu, H., Caldara, R., Schyns, P.G.: Facial expressions of emotion are not culturally universal. Proc. Natl. Acad. Sci. **109**(19), 7241–7244 (2012)
18. Khare, S.K., Blanes-Vidal, V., Nadimi, E.S., Rajendra Acharya, U.: Emotion recognition and artificial intelligence: a systematic review (2014–2023) and research recommendations. Inf. Fusion 102019 (2023)
19. Kosti, R., Alvarez, J.M., Recasens, A., Lapedriza, A.: Emotion recognition in context. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1667–1675 (2017)

20. Kosti, R., Alvarez, J.M., Recasens, A., Lapedriza, A.: Context based emotion recognition using EMOTIC dataset. IEEE Trans. Pattern Anal. Mach. Intell. **42**(11), 2755–2766 (2019)
21. Lee, J., Kim, S., Kim, S., Park, J., Sohn, K.: Context-aware emotion recognition networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 10143–10152 (2019)
22. Li, S., Deng, W.: Deep facial expression recognition: a survey. IEEE Trans. Affect. Comput. **13**(3), 1195–1215 (2020)
23. Li, Y., Aizawa, K., Matsui, Y.: Manga109dialog a large-scale dialogue dataset for comics speaker detection. preprint arXiv:2306.17469 (2023)
24. Louis, J.B., Burie, J.C.: Detection of buried complex text. Case of onomatopoeia in comics books. In: Coustaty, M., Fornés, A. (eds.) ICDAR 2023. LNCS, vol. 14193, pp. 177–191. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41498-5_13
25. Matsui, Y., et al.: Sketch-based manga retrieval using manga109 dataset. Multimed. Tools Appl. **76**(20), 21811–21838 (2017)
26. Mittal, T., Guhan, P., Bhattacharya, U., Chandra, R., Bera, A., Manocha, D.: Emoticon: context-aware multimodal emotion recognition using Frege's principle. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14234–14243 (2020)
27. Nguyen, N.-V., Rigaud, C., Burie, J.-C.: Comic characters detection using deep learning. In: 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 3, pp. 41–46. IEEE (2017)
28. Nguyen, N.-V., Rigaud, C., Burie, J.-C.: Digital comics image indexing based on deep learning. J. Imaging **4**(7) (2018)
29. Nguyen, N.-V., Vu, X.-S., Rigaud, C., Jiang, L., Burie, J.-C.: ICDAR 2021 competition on multimodal emotion recognition on comics scenes. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12824, pp. 767–782. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86337-1_51
30. Pham, L., Vu, T.H., Tran, T.A.: Facial expression recognition using residual masking network. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 4513–4519. IEEE (2021)
31. Plutchik, R.: The Nature of Emotions: human emotions have deep evolutionary roots, a fact that may explain their complexity and provide tools for clinical practice. Am. Sci. **89**(4), 344–350 (2001)
32. Russell, J.: A circumplex model of affect. J. Pers. Soc. Psychol. **39**, 1161–1178 (1980)
33. Shan, C., Gong, S., McOwan, P.W.: Facial expression recognition based on local binary patterns: a comprehensive study. Image Vis. Comput. **27**(6), 803–816 (2009)
34. Théodose, R., Burie, J.C.: KangaiSet: a dataset for visual emotion recognition on manga. In: Coustaty, M., Fornés, A. (eds.) ICDAR 2023. LNCS, vol. 14193, pp. 120–134. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-41498-5_9
35. Wang, Z., Lao, L., Zhang, X., Li, Y., Zhang, T., Cui, Z.: Context-dependent emotion recognition. J. Vis. Commun. Image Represent. **89**, 103679 (2022)
36. Woo, S., Park, J., Lee, J.-Y., Kweon, I.S.: CBAM: convolutional block attention module. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19 (2018)

CHART-Info 2024: A Dataset for Chart Analysis and Recognition

Kenny Davila[1](✉), Rupak Lazarus[2], Fei Xu[3], Nicole Rodríguez Alcántara[4], Srirangaraj Setlur[3], Venu Govindaraju[3], Ajoy Mondal[2], and C. V. Jawahar[2]

[1] School of Computing, DePaul University, Chicago, IL 60604, USA
kdavilac@depaul.edu
[2] International Institute of Information Technology, Hyderabad, India
rupak.lazarus@research.iiit.ac.in, {ajoy.mondal,jawahar}@iiit.ac.in
[3] CSE, University at Buffalo, Buffalo, NY 14260, USA
{fxu3,setlur,govind}@buffalo.edu
[4] Facultad de Ingenieria, Universidad Tecnológica Centroamericana, Tegucigalpa, Honduras
nicole.rodriguez@unitec.edu

Abstract. Charts are tools for data communication used in a wide range of documents. Recently, the pattern recognition community has shown interest in developing methods for automatically processing charts found in the wild. Following previous efforts on ICPR's CHART-Infographics competitions, here we propose a newer, larger dataset and benchmark for analyzing and recognizing charts. Inspired by the steps required to make sense of a chart image, the benchmark is divided into 7 different tasks: chart image classification, chart text detection and recognition, text role classification, axis analysis, legend analysis, data extraction, and end-to-end data extraction. We also show the performance of different baselines for the first five tasks. We expect that the increased scale of the proposed dataset will enable the development of better chart recognition systems.

Keywords: Charts · Dataset · Graphic Recognition

1 Introduction

Charts are tools for data communication used in a wide range of documents. The pattern recognition community has displayed a significant interest in methods for analyzing and recognizing charts in the wild [8]. There are rules and regular patterns that define how data can be converted into charts of different types. Yet, current models still struggle with complex graphics, especially when these do not adhere strictly to these rules and conventions. In contrast, humans can still successfully interpret the data encoded in these images.

Recent years have seen the development of very deep networks which can solve a variety of tasks as long as they are trained with enough data. For chart

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 297–315, 2025.
https://doi.org/10.1007/978-3-031-78495-8_19

recognition, most available large-scale datasets are synthetic, often created using an artificial process to generate charts based on data from real sources [10,32]. However, users can employ many tools to create charts with diverse visual styles. Often, they do not just convert data tables arbitrarily into charts, but instead use domain knowledge to choose appropriate chart types and conventions and carefully communicate their message [8]. Therefore, using a single tool to create large synthetic datasets does little to capture the diversity of charts in the wild.

Large-scale datasets are also needed to evaluate chart recognition systems. The CHART-Infographics competitions [9–11] were proposed with the goal of becoming the go-to chart recognition benchmark. Earlier editions included synthetic datasets, but systems trained only on synthetic charts performed very poorly on real charts seen in documents in the wild. Therefore, recent editions of CHART-Info have focused on providing large chart datasets based on real charts. This work presents the next iteration of this effort, the *CHART-Info 2024* dataset, which provides a major increase in the amount of training data facilitating training of larger models. At the same time, we provide a brand new test set with non-disjoint splits to evaluate each task using far more images.

CHART-Info defines six functional tasks critical to the chart recognition process: Chart Image Classification (Task 1), Text Detection and Recognition (Task 2), Text Role Classification (Task 3), Axis Analysis (Task 4), Legend Analysis (Task 5), and Data Extraction (Task 6). To facilitate the development of task-specific models, each task receives as input the ideal outputs from some of the previous tasks. An additional task is formulated for end-to-end data extraction (Task 7), where only the chart image is provided. Tasks 6 and 7 must produce an approximation of the data table used to create the chart. In this work, we provide baselines for all tasks except 6 and 7. These baselines are based on open-source models, but multiple domain adaptations and heuristic rules have been used to make them work well on chart-specific tasks. The chart images, annotations, evaluation tools and custom baseline code can be found here: https://github.com/kdavila/CHART_Info_2024.

2 Related Works

Multiple datasets have been created for different chart-related tasks. Earlier datasets were created by collecting chart images using web engines and manual filtering [5,34]. In this category we find the Revision dataset [34] (2,500 images of 10 chart types) and the dataset by Chagas et al. [5] (4,837 images of 10 chart types). However, web images are often available under restrictive licenses.

Other works have collected charts directly from data-oriented web sources. ExcelChart400k [30] was created by collecting Excel spreadsheets from the web. The dataset contains a total of 386,966 charts extracted from these spreadsheets along with the tabular data used to create them (no manual annotation required). Nevertheless, all charts are created using a single tool, thus they lack visual diversity. The chart text was also replaced with random characters, affecting the ability to incorporate multi-modal methods that use text. Another example is the ChartQA [31] dataset which has 21,945 charts (mostly in vector format) of 3 types (bar, line and pie), extracted from 4 websites. Annotations for visual question answering (VQA) tasks were collected via crowdsourcing.

Some synthetic chart datasets have been proposed for tasks such as data extraction [3,10] and VQA [22,32]. Some important advantages of generating synthetic data include: better scalability, cleaner and more detailed annotations, and relatively low cost. The AdobeSynth dataset [10] has 202,550 chart images (10 classes) generated with Matplotlib using data from multiple web sources. Bajić and Job [3] used Plotly to create a dataset with over 120K images from 20 chart classes. The DVQA [22] dataset contains 300K images of synthetic bar charts generated with the matplotlib library. The PlotQA [32] dataset has 224,377 images (bar, scatter, and line plots) generated using an unspecified tool. DVQA and PlotQA provide millions of question-answer pairs.

Some recent efforts, including this work, have extracted and manually annotated images from scientific literature to create large-scale datasets. These have the advantage of being more reliably available than random images from the web, and many of them are available under licenses that allow their redistribution and even commercial usage. Two examples are the FigureSeer [36] dataset (60K images from 20K papers) and the DocFigure [20] dataset (33K images), but these are mostly designed for figure type classification (including many charts). The CHART-Infographics competitions have led to the creation of datasets with increasing scales. The dataset presented here is an major extension of our previous dataset from ICPR CHART-Info 2022 [11], which is based on real charts extracted from PubMed Central (PMC).

3 The CHART-Info 2024 Dataset

This work extends previous efforts from the CHART-Infographics competitions [9–11]. Every edition has provided a novel test dataset based on real charts. While the first edition provided AdobeSynth for training, the second edition [9] expanded the previous test dataset to create the first training dataset based on real charts. The third [11] merged previous datasets to form the new training dataset. The training dataset of CHART-Info 2024 merges previous [11] training and testing datasets, and provides the largest test dataset so far (See Table 1).

The novel test dataset was created using a similar methodology to the previous edition [11]. We selected papers added to the Open Access Section of the PMC between Dec. 2017 and Oct. 2021. We only considered papers released under *CC BY* or *CC-0* licenses containing the keywords “chart” or “plot” in their main text. Out of 241,396 papers matching these filters, we randomly selected 20K papers that were not already included in earlier versions of our dataset.

A binary image classifier was used to identify chart candidates from all the figures in these papers. Then, these candidates were manually classified by chart type. Note that all images in our dataset are in JPEG format, just as provided by the PubMed Central. As a result, our dataset has the limitation of not including

Table 1. Distribution of chart types on the training and testing datasets. Values are compared against earlier versions of the dataset.

Chart Type	ICPR 2020 [9]		ICPR 2022 [11]		ICPR 2024	
	Train	Test	Train	Test	Train	Test
Area	120	52	172	136	308	229
Line	7,401	3,155	10,556	3,400	13,955	5,142
Manhattan	123	53	176	80	256	68
Scatter	875	475	1,350	1,247	2,597	1,311
Scatter-Line	1,260	558	1,818	1,628	3,446	1,684
Pie	170	72	242	191	433	213
Vertical Box	316	447	763	775	1,538	802
Horizontal Bar	429	358	787	634	1,421	636
Vertical Bar	3,818	1,636	5,454	3,745	9,199	3,692
Horizontal Interval	109	47	156	430	586	326
Vertical Interval	342	147	489	182	671	202
Map	373	160	533	373	906	363
Heatmap	138	59	197	180	377	177
Surface	110	45	155	128	283	127
Venn	52	23	75	131	206	121
Total	15,636	7,287	22,923	13,260	36,182	15,093

Table 2. Statistics (min., median, max.) for different image attributes in our dataset.

Dataset	Image Width			Image Height			Image DPI			File Size		
	Min	Med	Max	Min	Med	Max	Min	Med	Max	Min	Med	Max
Training	76	709	10,800	24	486	6,000	28	600	2,400	2,577	49,999	7,986,057
Testing	118	714	6,299	66	490	7016	72	300	2400	3,036	54,506	2,859,605

any images in vector formats. However, we note that all images in vector format can be easily rasterized to be recognized using models trained with raster images.

The original creators of the images in our dataset used a variety of tools to make them leading to diverse quality and sizes as shown in Table 2. Training set images go from 76×75 to $10,800 \times 6,000$ pixels, while testing set images go from 120×66 to $6,299 \times 6,299$ pixels. This is a challenge for vision models expecting fixed resolutions. For tasks such as image classification, it might be better to downsize large images, but details that help to differentiate between challenging pairs (e.g. line vs. scatter-line) might be lost when the images are shrunk (see Fig. 2.b.). For other task such as text recognition, higher resolution images are better because the text is more readable. However, the variability in the relative scales of text and other chart objects can make detection tasks harder.

Table 3. Available charts for training and testing per task.

Dataset	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7
Training	36,182	8,343	8,343	6,965	7,065	5,427	5,427
Testing	15,093	3,280	3,280	3,128	3,128	2,676	2,676

Data annotation was a collaborative effort between teams at 3 universities: University at Buffalo (USA), IIIT-Hyderabad (India), and UNITEC (Honduras). At each location, the annotators collected the ground truth (GT) in 3 sequential stages: *Image class* annotation, *Text* annotation, and *legend, axes and data* annotation. At every stage, annotators were assigned images in batches, and used our publicly available tools[1] to annotate them. The first two stages were semi-automatic, because recognition systems were used to generate initial labels. Then, the annotators had to verify and correct these labels, effectively cutting down the annotation time in half compared to previous years. For quality control, every annotation at every stage had to be approved by one validator who enforced different rules to achieve consistent annotations.

Table 3 shows the number of charts available for training/testing per task. Because of the competition context, previous CHART-Info testing datasets used 5 disjoint splits to evaluate different tasks [9,11]. However, CHART-Info 2024 is an offline evaluation benchmark, eliminating the need for disjoint splits, providing far more data to evaluate each task. Based on the availability of GT per chart, we propose 4 non-disjoint splits for training and testing specific tasks: *DS1*, *DS2*, *DS3* and *DS4*. Split *DS1* is basically the entire dataset (column *Task 1* in Table 3), which can only be used for Task 1. Split *DS2* represents all charts that have GT for task 2 (column *Task 2* in Table 3), used for task 2 only. Tasks 3, 4 and 5 share the same inputs, and Split *DS3* only considers the charts that have GT for the three tasks: 6,957 for training and 3,128 for testing. Finally, Split *DS4* includes all fully annotated charts that can be used to evaluate Tasks 6 and 7 (column *Task 6* in Table 3). All results presented in Sect. 4 are based on the following protocol: The training portion of each data split is further divided into 80% for training and 20% for validation. Then, the test portion of the same data split is used for evaluation.

4 Tasks and Baselines

In this section, we will describe the inputs, outputs, evaluation metrics and baselines for each task supported by CHART-Info 2024. Based on existing chart recognition systems [8], these tasks are defined in a sequence: chart image classification (Sect. 4.1), detection and recognition of text (Sect. 4.2), text role classification (Sect. 4.3), axes analysis (Sect. 4.4), legend analysis (Sect. 4.5), and chart data extraction (Sect. 4.6). Alternatively, one can design systems that handle the whole process in a end-to-end manner (Sect. 4.7).

[1] https://github.com/kdavila/ChartInfo_annotation_tools.

4.1 Task 1. Chart Image Classification

Task Description. The type of the chart in an image determines the rules used to interpret the data encoded in its graphical elements. Therefore, the first task is chart image classification, where every image in the dataset belongs to one of the classes listed in Table 1. We focus on chart types that are well represented in our data source, and some of these might be known under different names.

Inputs, Outputs and Metrics. The input is an image, and the output is a chart class. Evaluation is based on standard classification metrics such as the per-class recall, precision and F1 score. Table 1 shows that this is an unbalanced dataset, therefore we use the macro-average of the F1 scores for all classes as the final score to emphasizes the importance of correctly identifying all classes.

Baselines. We used image classification methods which have achieved good results on chart images in the past. We obtained the following results: ResNet-18 [16] (91.20%), ResNet-32 [16] (91.17%), ResNet-50 [16] (92.37%), Inception V3 [37] (92.07%), Xception [7] (92.98%), MobileNet-V3-Small [18] (89.30%), MobileNet-V3-Large [18] (91.40%), EfficientNet-B0 [38] (91.59%), EfficientNet-B1 [38] (92.14%), Swin-Tiny [29] (91.02%), and Swin-Base [29] (**93.60**%). The highest score is achieved by the more recent model based on transformers [29]. Nevertheless, earlier models such as ResNet-50 [16] still achieve competitive results, with F1 scores only 1.23% lower than the best model. This task can be hard for many reasons [39], and these numbers show that better models are still needed. Some images are hard to classify even for human annotators, like the scatter-line shown in Fig. 2.b which can be easily confused with a line chart.

4.2 Task 2. Chart Text Detection and Recognition

Task Description. Text is an important component of the semantics of a chart image. The goal of this task is to detect and recognize individual text blocks in the image. Unlike many scene text detection benchmarks, we are not interested in detecting isolated words, but rather we need to identify text blocks meant to be interpreted as a single unit (e.g. an axis title, a legend entry, etc.).

Inputs, Outputs and Metrics. The inputs are the chart image and its classification. The output is a list of text regions (*detection*), represented by quadrilaterals, and their corresponding transcriptions (*recognition*), represented by strings which might include LATEX notation to handle special symbols and formulas found in charts. For each image, the *predicted texts* are matched against the *GT texts* if their $IoU \geq 0.5$, but a 1-to-1 matching constraint is enforced. The IoU values of matching texts are summed and the total is divided by $max(|predicted\ texts|, |GT\ texts|)$ to produce the *per-image IoU* score. For every GT text, we compute the normalized character error rate (NCER) between its transcriptions and the text of its corresponding prediction. Unmatched GT texts receive a NCER score of 0. All NCER values are summed and divided by $|GT\ texts|$ to produce the *per-image NCER* score. *Detection* and *Recognition* results are evaluated using the average of the *per-image IoU* and *per-image*

NCER scores, respectively. The final metric for Task 2 is the harmonic mean (f-score) of the *detection* and *recognition* scores.

Baselines for Detection. On average, charts have simpler backgrounds than natural scenes, but text regions can be really small, and they can overlap other graphical elements. One missing text region can have a huge impact in the overall accuracy of the whole chart recognition process. While scene text detectors often target isolated words, text in charts is better analyzed using coherent text regions (e.g. a complete data series name). This is hard considering that text regions in charts can go from one symbol to multiple lines of text.

We first considered two out-of-the-box baselines, Tesseract OCR engine v5.3.1 [1] and PaddleOCR engine v3 (PPv3) [24], which achieved average IoU scores of 0.3035 and 0.6022, respectively. These results show that, out of the box, they do not work well on charts probably because our target is text blocks which is not what these methods produce by default. These results already include some post-processing rules that helped but by only so much.

We then considered multiple baselines retrained with our data (*DS2*). Some of these models are based on the PPv3 framework [24], and these are pretrained on the ICDAR 2015 Scene Text Detection dataset [23]. We also considered models from the MMRotate framework [45], which were designed for rotated object detection in aerial images, and are pretrained on ImageNet-1k. We acknowledge that the accuracy of third-party re-implementations might be slightly different to the original models. Table 4 shows results for text detection baselines retrained on our dataset. In the case of PPv3 [24], configurations using the larger ResNet-50 [16] backbone achieved better results than their counterparts using MobileNet-V3 [18], although by a small margin in many cases. Surprisingly, while DB++ [27] has produced better results than EAST [44] on natural scenes, EAST achieved better results here. Nevertheless, the best results were obtained using RoI Transformer [12] with Swin-Tiny [29] backbone, from the MMrotate framework [45]. This model might be the best in handling rotated text.

Baselines for Recognition. To make a fair comparison between text recognition baselines, we apply each recognition algorithm over text detection GT. During training and evaluation, the unicodeit Python library is used to convert the special LATEX annotations into Unicode symbols. This library cannot handle all special symbols and formulas, but is appropriate for the vast majority of alphanumeric strings. We created a custom dictionary based on the 250 most common Unicode symbols in the training set to retrain some of our baselines.

Text recognition models often expect the inputs to contain a single horizontal line of text with at most so many characters. They do not work well with images of texts that are very long, rotated and/or multi-line. Multi-line text blocks are rare in our dataset, but long texts are very common. We use simple rules to identify long and/or multi-line text candidates, and then apply greedy algorithms to cut the image horizontally and/or vertically as required. The recognizer is used over each partition, and the results are concatenated.

Table 4. Baselines for Chart Text Detection. Backbones include MobileNetv3 [18] (MN3), ResNet-50 [16] (RN50), ReResNet-50 [15] (RRN50) and Swin Tiny [29] (ST).

Method	IoU
DB [26] - MN3	0.7809
DB [26] - RN50	0.7866
PSENet [40] - MN3	0.8242
PSENet [40] - RN50	0.8263
DB++ [27] - RN50	0.7996
EAST [44] - MN3	0.8036
EAST [44] - RN50	0.8396
ReDet [15] - RRN50	0.8875
RoI Trans. [12] - RN50	0.8828
RoI Trans. [12] - ST	**0.8890**

Table 5. Baselines for Chart Text Recognition. These baselines are based on the PPv3 framework, were pre-trained on ICDAR 2015 [23] and retrained on our dataset. Results are provided using *Line Splitting* (w LS) and *Without Line Splitting* (w/o LS).

Method	w/o LS NCER	w LS NCER
SAR [25]	0.9064	0.9202
SRN [42]	0.9098	0.9267
RobustScanner [43]	0.9133	0.9312
VisionLAN [41]	0.9132	0.9316
CRNN [35]	**0.9480**	0.9477
StarNet [28]	0.9293	0.9480
RFL [19]	0.9329	0.9506
ABINet [14]	0.9360	0.9528
SVTR [13]	0.9323	**0.9549**

Table 6. Baselines for Chart Text Detection and Recognition. We consider some Out-of-the-box (OOB) configurations, and the best configurations reported earlier.

Detection Method	Recognition Method	OOB	IoU	NCER	H-Mean
Tesseract [1]	Tesseract [1]	Yes	0.3035	0.3663	0.3320
PPv2 [24] DB [26]	PPv2 [24] CRNN [35]	Yes	0.5688	0.7074	0.6305
PPv3 [24] DB [26]	PPv3 [24] SVTR [13]	Yes	0.6022	0.7866	0.6821
RoI Trans. [12] Swin Tiny [29]	PPv3 SVTR [13]	No	**0.8890**	**0.9264**	**0.9073**

Rotated text is common in charts, specially on *axis titles* and *tick labels*, and can be long and/or multi-line as well. All text regions (any rotation) are always projected into axis-aligned rectangular regions, which can only have 0, +90, +180 or −90 degree rotations. The PPv3 framework includes an angle classifier but it did not perform well on chart text. Because of this, we used simple rules based on recognition confidence scores to simultaneously handle rotated text and long and/or multi-line text. When needed, this method tests multiple combinations of rotations with image splitting, and keeps the most confident transcription.

We considered three out-of-the-box baselines, Tesseract OCR [1], PPv2 [24] with CRNN [35], and PPv3 [24] with SVTR [13], which achieved NCER scores of 0.8483, 0.8226 and 0.8921, respectively. These baselines do not use any of our rules to handle special cases. We then experimented using our dataset to retrain multiple models from the PPv3 framework, and applying our rules to handle multi-line and rotated text. Table 5 shows the results for these models, considering whether the horizontal line splitting algorithm (LS) was used to deal with long text candidates or not. We found CRNN [35] to be the most robust in terms of handling long texts on its own. However, by using our horizontal line splitting algorithm, other methods achieved higher recognition results. The best recognition method was SVTR [13].

Complete Baselines. Table 6 shows the results for end-to-end chart text detection and recognition. Here, recognition results are affected by errors made by the detection model. We only consider out-of-the-box baselines, and the combination of the best models for detection (ROI Trans [12] with Swin-Tiny [29]) and recognition (PPv3 with SVTR [13]). This combined model is also using all of the intermediate rules used for handling rotated, multi-line and/or long texts.

4.3 Task 3. Chart Text Role Classification

Task Description. This task aims to determine the role or function that each text region has on the chart. Our dataset considers 9 roles: *chart title*, *axis title*, *tick label*, *tick grouping*, *legend title*, *legend label*, *value label*, *marker label*, and *other*. This covers the categories needed to make sense of a chart image. The *other* category is used to group additional less common roles. These roles are illustrated in Fig. 1.

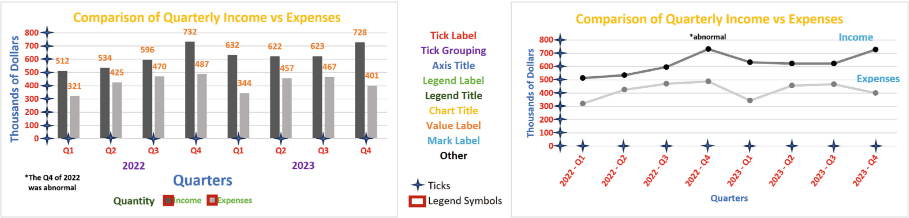


Fig. 1. Targets for multiple tasks. Different text colors are used to illustrate our text roles in two charts (Task 3). We also show the expected ticks using stars (Task 4), and red rectangles define the legend symbols (Task 5). Best seen in digital format. (Color figure online)

Inputs, Outputs and Metrics. The inputs include the chart image and the GT outputs for Tasks 1 and 2. The expected output is a list of the roles for each GT text region. Like Task 1, evaluation is based on classification metrics.

Baselines. This task is similar to general object classification on images. However, two identical objects (text regions) can have different classes (roles) based on their position within the chart layout. Class imbalance also makes this task challenging, where *tick labels* make about 70.11% of all text regions in the training dataset, while *legend title*, *chart title*, and *tick grouping* are so rare that even combined represent just 1.15% of all text regions. However, all classes have the same impact on the final metrics.

Table 7. Baselines for Text Role Classification. Columns are F1 scores (%) for *tick label* (TL), *axis title* (AT), *legend label* (LL), *value label* (VL), *legend title* (LT), *mark label* (ML), *tick grouping* (TG), *other* (O), *Chart Title* (CT) and their macro average. For each baseline, we consider the V35 (†) and V345 (‡) variations.

Method	TL	AT	LL	VL	LT	ML	TG	O	CT	AVG
†ReDet [15] (R50 [16])	98.8	97.1	97.9	81.2	79.1	54.4	55.7	71.1	83.2	79.85
†RoI Trans [12] (R50 [16])	98.6	97.3	97.8	81.3	78.0	60.5	54.4	71.0	82.6	80.18
†RoI Trans [12] (Swin Tiny [29])	98.9	98.0	98.1	84.2	79.6	62.1	60.7	74.9	87.9	82.72
†YOLO-V8x [21]	98.8	98.1	97.3	84.7	85.3	63.0	70.7	74.5	89.9	84.71
†Deformable DETR [46]	99.4	98.6	98.6	83.8	87.1	63.8	70.6	75.9	91.3	85.45
†C. Mask R-CNN (Swin Base) [29]	99.2	98.7	98.6	85.4	89.6	68.7	70.9	77.3	92.6	**86.78**
‡ReDet [15] (R50 [16])	98.9	96.7	98.0	81.3	79.6	56.2	57.5	72.2	83.1	80.40
‡RoI Trans [12] (R50 [16])	98.7	97.4	98.1	81.3	75.7	60.2	56.3	71.8	82.6	80.22
‡RoI Trans [12] (Swin Tiny [29])	99.0	98.0	98.1	83.7	81.5	63.4	61.2	74.5	88.5	83.10
‡YOLO-V8x [21]	98.5	96.8	96.8	79.3	76.0	51.9	59.9	67.7	72.7	77.76
‡Deformable DETR [46]	99.4	98.6	98.8	84.2	87.4	66.3	74.9	75.9	92.1	86.40
‡C. Mask R-CNN (Swin Base) [29]	99.2	98.7	99.0	85.2	89.6	69.4	69.8	77.5	90.8	86.57

We created our baselines for task 3 by training different object detector models using the role of each text region as their target class. We create variations of these models to simultaneously deal with multiple tasks. The first, *V35*, deals with tasks 3 and 5, and uses the original 9 text roles. The second, *V345*, deals with tasks 3, 4 and 5, and needs 12 roles for text. This is because it replaces the *tick label* class with 4 per-axis classes (more details in Sect. 4.4).

Task 3 requires assigning classes to text regions in the GT, but the predictions made by the object detectors might not align with the GT. Overlapping pairs of GT text regions and predictions are scored using the harmonic mean of their IoU and prediction confidence. We then greedily pick the highest scoring matches while enforcing a 1-to-1 matching constraint. The class of the prediction is finally assigned to each of the matched GT text regions. Unmatched predictions are simply ignored, and unmatched GT text regions are omitted.

Table 7 shows the results for Task 3. The *tick label*, *legend label* and *axis title* classes, which represent 87.16% of the training dataset, have very high F1 scores. Meanwhile, the *mark label* and *tick grouping* classes, which represent only 1.98% of the training dataset, have the lowest F1 scores. Except for YOLO-V8 [21], most models have very similar results for both variations. The Cascade Mask R-CNN model with Swin-Base transformer [29] and Deformable DETR [46] are consistently the strongest model from this set.

4.4 Task 4. Chart Axis Analysis

Task description. Axes in charts define the space of the chart data. The goal of this task is to locate the main chart axes (horizontal and vertical), and then

Table 8. Baselines for Axes Analysis. All of them are based on Variation 345

Method	Rec. (%)	Prec. (%)	F1 (%)
ReDet [15] (R50 [16])	83.89	85.72	84.79
RoI Trans [12] (R50 [16])	84.09	85.27	84.67
RoI Trans [12] (Swin Tiny [29])	84.56	86.08	**85.31**
YOLO-V8x [21]	54.35	56.64	55.47
Deformable DETR [46]	**85.38**	85.18	85.28
Cascade Mask R-CNN (Swin Base) [29]	77.32	**86.46**	81.63

link specific points in the axes (ticks) with text (tick labels). This location should be independent of the existence of visual tick marks.

Inputs, Outputs and Metrics. Inputs are the same as Task 3. The output is a dictionary organizing the tick positions by axis. Then, per axis, a set of pairs (text id, point) is expected. Each pair represents a *tick label* by their unique id in the GT, and the point represents the tick position.

Evaluation considers precision and recall of predicted ticks on the main axes (x-axis at the bottom and y-axis at the left). Secondary axes (top or right) are ignored. Predicted and GT ticks are matched by text id, and each match is weighted based on the distance between the GT location and the predicted location. First, the distances are normalized by the length of the image diagonal, and matches with distance $\geq$0.02 receive a weight of 0, and distance $\leq$0.01 receive a weight of 1. For 0.01 $<$ distance $<$ 0.02, an interpolated weight between 1 and 0 is used. Missing ticks and ticks associated with the wrong axis have weights of 0. The total weight of all matches is divided by the number of GT ticks to compute recall and by the number of predicted ticks to compute precision. Then, the overall recall and precision metrics are the macro averages of the per-axis values. Finally, we compute F1 score as the final per-chart score, and the average of the per-chart scores are computed for the entire evaluation set.

Baselines. While the GT text regions are known for this task, their corresponding roles are unknown. Because of this, our baselines work in combination with role predictions from Task 3, to identify all *tick labels*. The next challenge is to associate these to their corresponding axes. We use Variation V345 (see Task 4.3) which refines the *tick label* class by directly predicting if the region is a *tick label* of: *x-axis* (bottom), *y-axis* (left), *x2-axis* (top) or *y2-axis* (right).

The next step is to associate the *tick labels* to specific image locations. A common idea is to detect *tick marks* and use rules to match these to *tick labels*, but in many cases the *tick marks* are not visible or do not correspond to positions that should be associated with *tick labels*. To solve this problem, we add an *axes corner* object, which is a box of 10-by-10 pixels, centered at the origin of both x and y axes (bottom-left corner). The center of this box, (c_x, c_y), provides the coordinates shared by all ticks in a given axis (c_x for y axis, c_y for x axis). We use rules to determine the other coordinate using the rotated bounding box of the corresponding *tick label*. In most cases we simply use the center of the *tick label* bounding box: vertical center for y axis, horizontal center for x axis. Rotated *tick labels*, commonly found in the x axis, are the exception to this, and we use the x coordinate of the top-most point in their rotated bounding box.

Table 8 shows the results for this task. The performance for most object detectors is reasonably good considering that they do not detect visual tick marks. The baselines will fail when the *axes corner* box is incorrectly detected, or not detected at all. If multiple boxes are detected, the most confident is picked, but that can also produce incorrect outputs. Errors made in Task 3 (e.g., false positives/negatives for *tick labels*) are also propagated here. Also, *tick labels* associated with the wrong axis reduce precision for one axis, and recall for the other. Here, the ROI Trans model [12] achieved the highest score, with more consistent recall and precision levels than other models. The second best is Deformable DETR [46].

4.5 Task 5. Chart Legend Analysis

Task Description. A legend is made by a set of *legend entries*, which are (*legend label*, *legend symbol*) pairs. Each *legend label* usually corresponds to one specific data series in the chart. The *legend symbol* exemplifies the appearance of the corresponding *data marks*. An example is shown in the left side of Fig. 1. The goal of this task is to identify the *legend entry* pairs in the image.

Inputs, Outputs and Metrics. Inputs are the same as task 3. The output is a list of *legend entry* pairs (text id, bounding box), where the text id represents a *legend label*, and bounding box represents the associated symbol.

The evaluation of this tasks requires correct pairings between *legend labels* and *legend symbols*. For a given chart, predicted *legend entries* are initially matched to GT by the id of the *legend labels*. For each matching pair, the area of intersection between the GT *legend symbol* and the predicted one is computed and used to get two metrics: an IoU-based score (divide by the area of the union) and a recall-based score (divide by the area of the GT bounding box). The sum over all *legend entries* is computed for both scores, and then they are divided by the maximum between the number of GT *legend entries* and predicted *legend entries*. Finally, the average over the evaluation set is computed for both metrics.

Many charts have rather small and thin *legend symbols* (e.g., height of 2 pixels). IoU-score can be over-punishing on bounding boxes that correctly capture the *legend symbol*, but are slightly thicker. The recall-based score is also considered here because of this.

Baselines. Similar to Task 4, this task has access to the GT text regions, but the roles are unknown. All *legend labels* and *legend symbols* candidates need to be identified, and these need to be combined into *legend entry* pairs. As described before, we consider object detection baselines that combine multiple tasks on the same network. For task 5, we simply add the *legend symbol* objects. The same network will produce all *legend label* and *legend symbol* candidates.

Table 9. Baselines for Legend Analysis. We consider variations V35 (†) and V345 (‡).

Method	Average BBox	
	IoU (%)	Recall (%)
†ReDet [15] (R50 [16])	83.09	95.33
†RoI Trans [12] (R50 [16])	83.41	95.62
†RoI Trans [12] (Swin Tiny [29])	**84.31**	95.56
†YOLO-V8x [21]	43.86	49.53
†Deformable DETR [46]	80.52	88.53
†Cascade Mask R-CNN (Swin Base) [29]	84.23	93.56
‡ReDet [15] (R50 [16])	83.66	95.57
‡RoI Trans [12] (R50 [16])	83.26	**95.69**
‡RoI Trans [12] (Swin Tiny [29])	83.84	95.42
‡YOLO-V8x [21]	43.42	49.31
‡Deformable DETR [46]	82.86	91.81
‡Cascade Mask R-CNN (Swin Base) [29]	84.12	93.62

The next challenge is to pair the candidates while considering false positives and negatives for both classes. Algorithms typically used for 1-to-1 matching in bipartite graphs will fail due to the noisy predictions. A simple approach is to pair each *legend symbol* with its closest *legend label*, but the way in which the distances are measured determines the quality of results. Based on the observation that for most charts, all *legend entries* have their symbols on the same side, we first estimate if all legend symbols in the image are left, right, above or below their labels. This direction is used to pick the corresponding edges of the bounding boxes of the legend labels, and their middle points are used as reference points for the labels. We then measure the distances between the reference points and the centers of the bounding boxes of the symbols. Matches are then sorted by increasing distance, and they are greedily picked in that order. Only matches between previously unmatched elements are accepted, and the process stops as soon as the first match involving a symbol or label previously matched appears. We do this to prevent spurious matches involving false positives/negatives based on the observation that valid legend entries in the same chart usually have similar edge distances between their symbols and labels.

Table 9 shows the results for this task. The recall-based scores show that most legend symbols are being detected and matched correctly, and that predicted boxes greatly overlap the symbol regions. It is possible that many of these predictions have the wrong thickness, leading to a much lower IoU-based scores in comparison. Errors can come from false positives/negatives of *legend symbols* and *legend label*. The method with the highest average F1 score for role classification achieves 98.6 F1 score for the *legend label* class (see Table 7). Therefore, it is likely that most errors come from failures to correctly detect the legend symbols. The baselines based on RoI Trans [12] consistently achieve some of the best scores for this task.

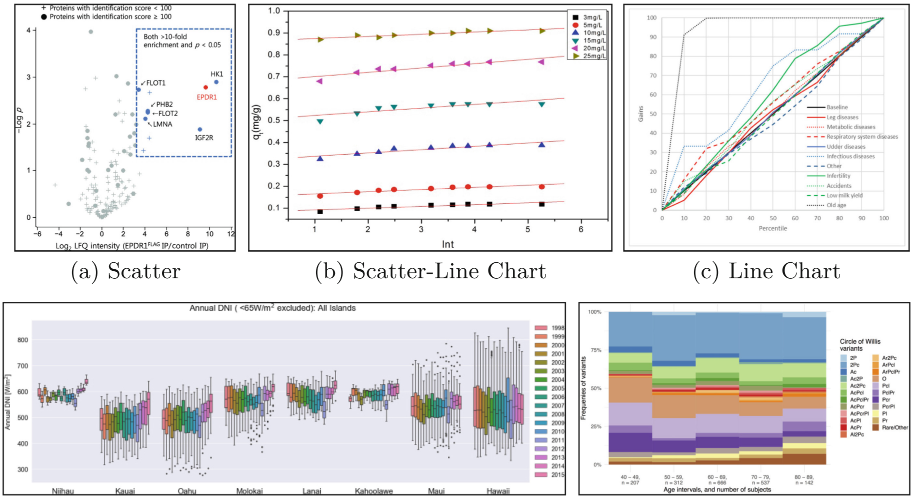
4.6 Task 6. Chart Data Extraction

Task Description. This task approximates the data table used to create a chart image. It is divided into two sub-tasks: Plot element detection and extraction (Task 6.a), and raw data extraction (Task 6.b). The first sub-task aims at correctly locating the data marks in the chart (e.g., lines in line charts, bars in bar charts, etc.). The second sub-task puts everything together (text, axes, legends, and data marks) to reconstruct the data encoded in the chart image. The chart type might be used to determine the right approach for this task.

Inputs, Outputs and Metrics. The inputs for this task are the outputs from all previous tasks (1–5). The outputs for Task 6.a depend on the chart type. Bar charts require a list of bounding boxes per bar. Line and Scatter plots require a list of points for each data series. Box plots require a tuple with the position of the components of each box (box top, box bottom, box median, top whisker, bottom whisker). For task 6.b, the output is a set of data series with name, and the list of data points (x, y) that make that data series, where x is the independent variable, and y is the dependent variable. Multiple metrics are considered depending on the type of chart (See [9,11] for details).

Baselines. The baselines for this task are complex, requiring combination of results from previous tasks. Existing works on chart recognition typically focus on a single chart type [8]. Meanwhile, our dataset provides data annotations for: horizontal/vertical bar charts, vertical box plots, scatter plots and line charts. Providing baselines for each of these chart types is out of the scope of this work. However, we briefly discuss the implications and complexities of creating data extraction models for these chart types.

Horizontal/Vertical Bar Charts. This is arguably the simplest chart type for data extraction. Standard object detectors often work with anchors of predefined aspect ratios, and bars in some charts can be extremely narrow or long. Most charts use bars of solid colors, but some charts use complex texture patterns instead. After detecting the bars, the next challenge is to infer the values represented by them. Every bar must be correctly mapped to a category (x value), a data series (e.g., “legend entry”), and an absolute value (y value). The orientation of the chart defines which axis is the x value (independent variable) and which one is the y value (dependent variable). Many charts use stacked and/or grouped bars, which require associating multiple bars to a single category (e.g., by proximity). Bars are usually associated to particular data series using legend analysis and their appearance. Finally, the y values of the bars are inferred through axes analysis and their spatial location. This process gets slightly more complicated for stacked bars where two extreme points are required to infer the value of each bar. Also, data points with y=0 often lead to *invisible bars*, but their existence might be inferred by analyzing the chart layout. Figure 2.e shows an example of a complex stacked vertical bar chart included in our dataset.



(a) Scatter (b) Scatter-Line Chart (c) Line Chart

(d) Vertical Box Plot (e) Stacked Bar Chart

Fig. 2. Examples of challenging charts in our dataset. (a) Scatter chart, extracted from [33]. (b) Scatter-line, extracted from [6]. (c) Line chart, extracted from [2]. (d) Vertical Box plot, extracted from [4]. (e) Stacked bar chart, extracted from [17].

Vertical Box Plots. The first step is to detect the boxes and their corresponding whiskers and median lines. Object detectors can be used to locate the boxes, but extreme aspect ratios can be a challenge. Whiskers and median lines might not be visible when their values are too close to the values represented by the top and/or bottom of the boxes. Similar to bar charts, we can find grouped box plots, where multiple boxes share a single category and each box needs to be linked to a particular legend entry. There are five points of interest in each box that need to be correctly mapped to their corresponding values in the y-axis. Figure 2.d shows a complex grouped vertical box plot included in our dataset. Note that horizontal box plots can be processed in a similar way, but we excluded them from our dataset because they are quite rare in practice. Also, many box plots include outliers represented by scatter marks, but we decided to currently exclude these from our benchmark because even human annotators often struggle to distinguish and recognize all individual points.

Scatter Plots. First, each data mark representing a data point must be detected. This might be easy when there are only a few large data marks, and extremely difficult when the plot region is cluttered with many small data marks. In the worst case scenario, we might only approximate the distribution of the original data used to create the plot. While our dataset includes all kinds of scatter plots, we only annotated data in cases where human annotators could accurately identify all individual data points. Data marks can be generated using all sorts of shapes, colors and sizes, making their automatic detection very challenging. After detection, each data mark needs to be mapped to a 2D point using their location and axis analysis. Legend analysis and the appearance of the data marks must be used to correctly map them to their corresponding data series. If no legend is present, then the appearance alone must be used to infer the existence of multiple data series. Figure 2.a shows an example of a complex scatter in our dataset where the colors on the legend do not match the colors in the plot region, and matching needs to be done using shapes, but this is also hard due to overlaps between data marks.

Line Plots. This is arguably the most challenging type here. Identifying the pixels of a given line is a segmentation problem which cannot be approached with basic object detectors. Some charts have solid colored lines which can be easily traced by basic segmentation algorithms. However, we find many charts with dashed lines and repeated line colors, where the thickness and/or dash patterns must be considered to differentiate them. Lines can also intersect and significantly occlude each other. Same as other chart types, lines must be associated with data series using legend analysis, but this can be difficult when the associated legend entries are very thin. Figure 2.c shows a line chart in our dataset which displays many of these issues. Some charts do not use legends, and instead directly provide names for each line using colored text within the plot region (e.g. *data mark label*) as illustrated on the right side of Fig. 1.

4.7 Task 7. End-to-End Data Extraction

The end-to-end data extraction task has the same goals, outputs and metrics as sub-task 6.b (Sect. 4.6). However, the only input is the image of a chart. This task was designed for systems that can handle the chart recognition process as a whole, removing the need to produce and evaluate intermediate outputs.

It is also possible to create a modular system which handles the recognition process using the individual tasks suggested in this benchmark. Not having any ground truth means that the real outputs from each module in the pipeline must be used. Any errors made in the earlier tasks will propagate to later tasks, affecting the overall recognition results.

Similar to Task 6, baselines for this complex task are out of the scope of this work.

5 Conclusion

In this paper, we have introduced the CHART-Info 2024 dataset, a natural extension to the existing ICPR 2022 CHART-Info datasets [11]. Apart from a brand new test dataset, we have also provided a considerable number of baselines for the first 5 tasks defined in our dataset. These baselines provide a starting point for researchers interested in chart recognition. We believe that our dataset will enable the development of better chart recognition systems which will be capable of dealing with complexities found on charts in the wild.

Acknowledgment. This material is based upon work partially supported by the National Science Foundation under Grant No. OAC/DMR 1640867.

References

1. Tesseract OCR – opensource.google.com. https://opensource.google.com/projects/tesseract
2. Adamczyk, K., Grzesiak, W., Zaborski, D.: The use of artificial neural networks and a general discriminant analysis for predicting culling reasons in Holstein-Friesian cows based on first-lactation performance records. Animals **11**(3), 721 (2021)
3. Bajić, F., Job, J.: Data extraction of circular-shaped and grid-like chart images. J. Imaging **8**(5), 136 (2022)
4. Bryce, R., Carreño, I.L., Kumler, A., Hodge, B.M., Roberts, B., Martinez-Anido, C.B.: Annually and monthly resolved solar irradiance and atmospheric temperature data across the Hawaiian archipelago from 1998 to 2015 with interannual summary statistics. Data Brief **19**, 896–920 (2018)
5. Chagas, P., et al.: Architecture proposal for data extraction of chart images using convolutional neural network. In: 21st IV, pp. 318–323. IEEE (2017)
6. Chen, J., Cai, Y., Clark, M., Yu, Y.: Equilibrium and kinetic studies of phosphate removal from solution onto a hydrothermally modified oyster shell material. PLoS ONE **8**(4), e60243 (2013)
7. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: Proceedings of CVPR, pp. 1251–1258 (2017)
8. Davila, K., Setlur, S., Doermann, D., Bhargava, U.K., Govindaraju, V.: Chart mining: a survey of methods for automated chart analysis. IEEE Trans. Pattern Anal. Mach. Intell. **43**(11), 3799–3819 (2020)
9. Davila, K., Tensmeyer, C., Shekhar, S., Singh, H., Setlur, S., Govindaraju, V.: ICPR 2020 - competition on harvesting raw tables from infographics. In: Del Bimbo, A., et al. (eds.) ICPR 2021. LNCS, vol. 12668, pp. 361–380. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-68793-9_27
10. Davila, K., et al.: ICDAR 2019 competition on harvesting raw tables from infographics (CHART-Infographics). In: ICDAR. IEEE (2019)
11. Davila, K., Xu, F., Ahmed, S., Mendoza, D.A., Setlur, S., Govindaraju, V.: ICPR 2022-challenge on harvesting raw tables from infographics. In: International Conference on Pattern Recognition. IEEE (2022)
12. Ding, J., Xue, N., Long, Y., Xia, G.S., Lu, Q.: Learning roi transformer for oriented object detection in aerial images. In: CVPR, pp. 2849–2858 (2019)
13. Du, Y., et al.: SVTR: scene text recognition with a single visual model. In: Raedt, L.D. (ed.) International Joint Conference on Artificial Intelligence, pp. 884–890. International Joint Conferences on Artificial Intelligence Organization (July 2022)
14. Fang, S., Xie, H., Wang, Y., Mao, Z., Zhang, Y.: Abinet: read like humans: autonomous, bidirectional and iterative language modeling for scene text recognition, pp. 7098–7107 (2021). https://arxiv.org/abs/2103.06495
15. Han, J., Ding, J., Xue, N., Xia, G.S.: Redet: a rotation-equivariant detector for aerial object detection. In: CVPR, pp. 2786–2795 (2021)
16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR, pp. 770–778 (2016)
17. Hindenes, L.B., Håberg, A.K., Johnsen, L.H., Mathiesen, E.B., Robben, D., Vangberg, T.R.: Variations in the circle of willis in a large population sample using 3d tof angiography: the tromsø study. PLoS ONE **15**(11), e0241373 (2020)
18. Howard, A., et al.: Searching for mobilenetv3. In: ICCV, pp. 1314–1324 (2019)
19. Jiang, H., et al.: Reciprocal feature learning via explicit and implicit tasks in scene text recognition (2021). https://arxiv.org/abs/2105.06229

20. Jobin, K., Mondal, A., Jawahar, C.: Docfigure: a dataset for scientific document figure classification. In: ICDARW, vol. 1, pp. 74–79. IEEE (2019)
21. Jocher, G., Chaurasia, A., Qiu, J.: Ultralytics YOLO, January 2023. https://github.com/ultralytics/ultralytics
22. Kafle, K., Price, B., Cohen, S., Kanan, C.: Dvqa: understanding data visualizations via question answering. In: CVPR, pp. 5648–5656 (2018)
23. Karatzas, D., et al.: ICDAR 2015 competition on robust reading. In: ICDAR, pp. 1156–1160. IEEE (2015)
24. Li, C., et al.: PP-OCRV3: more attempts for the improvement of ultra lightweight OCR system. arXiv preprint arXiv:2206.03001 (2022)
25. Li, H., Wang, P., Shen, C., Zhang, G.: Show, attend and read: a simple and strong baseline for irregular text recognition. ArXiv **abs/1811.00751** (2019)
26. Liao, M., Wan, Z., Yao, C., Chen, K., Bai, X.: Real-time scene text detection with differentiable binarization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 11474–11481 (2020)
27. Liao, M., Zou, Z., Wan, Z., Yao, C., Bai, X.: Real-time scene text detection with differentiable binarization and adaptive scale fusion. IEEE Trans. Pattern Anal. Mach. Intell. (2022)
28. Liu, W., Chen, C., Wong, K.Y.K., Su, Z., Han, J.: Star-net: a spatial attention residue network for scene text recognition. In: BMVC, vol. 2, p. 7 (2016)
29. Liu, Z., et al.: Swin transformer: hierarchical vision transformer using shifted windows. In: ICCV (2021)
30. Luo, J., Li, Z., Wang, J., Lin, C.Y.: Chartocr: data extraction from charts images via a deep hybrid framework. In: WACV, pp. 1917–1925 (2021)
31. Masry, A., Do, X.L., Tan, J.Q., Joty, S., Hoque, E.: ChartQA: a benchmark for question answering about charts with visual and logical reasoning. In: Findings of the ACL, pp. 2263–2279. Dublin, Ireland (May 2022)
32. Methani, N., Ganguly, P., Khapra, M.M., Kumar, P.: Plotqa: reasoning over scientific plots. In: WACV, pp. 1527–1536 (2020)
33. Park, J.K., et al.: Structures of three ependymin-related proteins suggest their function as a hydrophobic molecule binder. IUCrJ **6**(4), 729–739 (2019)
34. Savva, M., Kong, N., Chhajta, A., Fei-Fei, L., Agrawala, M., Heer, J.: Revision: automated classification, analysis and redesign of chart images. In: ACM Symposium on User Interface Software and Technology, pp. 393–402 (2011)
35. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2298–2304 (2017)
36. Siegel, N., Horvitz, Z., Levin, R., Divvala, S., Farhadi, A.: Figureseer: parsing result-figures in research papers. In: ECCV, pp. 664–680. Springer (2016)
37. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR, pp. 2818–2826 (2016)
38. Tan, M., Le, Q.: EfficientNet: rethinking model scaling for convolutional neural networks. In: ICML. PMLR, vol. 97, pp. 6105–6114. PMLR, 09–15 June 2019
39. Thiyam, J., Singh, S.R., Bora, P.K.: Chart classification: a survey and benchmarking of different state-of-the-art methods. IJDAR 1–26 (2023)
40. Wang, W., et al.: Shape robust text detection with progressive scale expansion network. In: CVPR, pp. 9336–9345 (2019)
41. Wang, Y., Xie, H., Fang, S., Wang, J., Zhu, S., Zhang, Y.: From two to one: a new scene text recognizer with visual language modeling network. In: ICCV, pp. 14194–14203 (2021)

42. Yu, D., Li, X., Zhang, C., Han, J., Liu, J., Ding, E.: Towards accurate scene text recognition with semantic reasoning networks. In: CVPR, pp. 12110–12119 (2020)
43. Yue, X., Kuang, Z., Lin, C., Sun, H., Zhang, W.: RobustScanner: dynamically enhancing positional clues for robust text recognition. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12364, pp. 135–151. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58529-7_9
44. Zhou, X., et al.: East: an efficient and accurate scene text detector. In: CVPR, pp. 5551–5560 (2017)
45. Zhou, Y., et al.: Mmrotate: a rotated object detection benchmark using pytorch. In: ACM International Conference on Multimedia (2022)
46. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable DETR: deformable transformers for end-to-end object detection. In: ICLR (2020)

AIO-HB: A Handwritten Text Image Dataset of Hindi and Bengali Indian Scripts for Handwritten Text Recognition

Piyush Kanti Samanta(✉) and Samit Biswas

Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, Shibpur, Howrah 711103, West Bengal, India
2020csp010.piyush@students.iiests.ac.in, samit@cs.iiests.ac.in

Abstract. Document Recognition and Analysis is a long-standing area of study that has been actively investigated for many decades. Handwritten text recognition (HTR) in Indian scripts is still in its early stages since the lack of publicly available datasets hampers the development of offline text recognition algorithms. Comparisons become tiresome due to the intricate structure of scripts, variances in depiction, and the presence of cursive writing. Hindi is the fourth most prevalent spoken language in the world, spoken by 615 million people, whereas Bengali is the sixth most popular, spoken by 265 million people [Source]. Both are read in a left-to-right direction. The accessible datasets of both languages are limited in size, have a limited number of writer's samples, and use limited annotations, which poses challenges in developing resilient solutions utilising contemporary machine learning methods. Here, we have prepared Hindi and Bengali text scripts, each covering all the letters of Hindi and Bengali literature, named AIO-HB(All in one Hindi-Bengali Dataset). The dataset, which can be accessed at https://sites.google.com/view/aio-hb-dataset, has also been made public for the benefit of the researchers. These handwritten scripts have been written by 202 different writers multiple times. These scripts are considered from individuals of different professions, including diverse ages and genders. The AIO-HB dataset is benchmarked using conventional deep-learning models for Handwritten Text Recognition (HTR). It can be used for various document image analysis applications, such as recognising script sentences, segmenting text lines, segmenting words, detecting words, and identifying writers. This article explores the reasons for improvements in HTR performance across scripts and the utility of annotation for Indian HTRs.

Keywords: Handwritten Dataset · Hindi and Bengali scripts · Handwritten Text Recognition · Benchmarking

1 Introduction

Since the beginning of the twentieth century, massive digitization initiatives have transformed paper documents and ancient historical manuscripts into digital

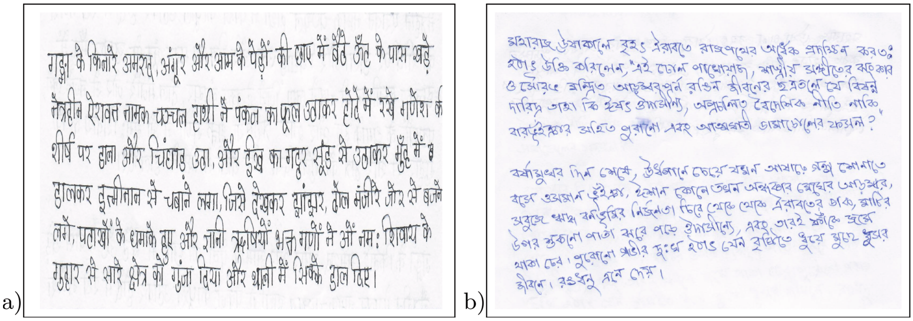
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 316–332, 2025.
https://doi.org/10.1007/978-3-031-78495-8_20

formats. Handwritten document image recognition is a demanding field within document image analysis, including tasks like text segmentation, written language recognition, writer identification, etc. Automatic language identification of paper-printed text pictures has obtained great success in this field with the help of the script-specific Optical Character Recognition model; however, the performance of recognising handwritten documents or text is hindered by variations in letters, cursive writing shapes, symbolic differences, and individual writing styles. Handwritten text recognition is an essential field in modern document analysis systems. It has been a prominent research area due to its wide range of possible applications, including postal automation, bank cheque processing, artificial information gathering, etc. Recent progress in text recognition systems has been propelled by the effectiveness of deep neural networks and annotated datasets. Standard datasets are essential for evaluating script recognition algorithms and enabling comparisons of results. Printed datasets are often used because of their ease of acquisition via newspapers, books, and the Internet. The fundamental challenge in creating a handwriting identification model is the wide range of handwriting styles and complex letter patterns within words. Another issue is the lack of standardised handwritten datasets in the public domain. The earlier maximum research was focused on recognising handwritten words in Latin [16], Arabic [15], Japanese, and Chinese [11] scripts. The IAM Handwritten Dataset [16], released more than 20 years ago, with the historic George Washington [23] and Botany Datasets [22], is well-known for handwritten text recognition in the Latin language. Currently, using these handwritten datasets, deep neural networks effectively provide significant analysis results through conventional feature-based approaches.

The Indian handwritten text recognition system is less explored than the above language-based handwritten text recognition model because of similar forms and traits resulting from their shared origin. Unlike many other regions globally, India uses diverse languages and scripts. Generally, deep neural networks need substantial training data to acquire new characteristics, making them suitable for swiftly processing a small number of handwritten texts. However, in the current scenario, the annotated datasets for Indian handwriting text recognition are small and narrow in focus. There are only a few publicly accessible datasets specifically dedicated to handwritten manuscripts. Thus, accumulating substantial handwritten datasets for various Indian scripts poses difficulties and high costs.

Hindi and Bengali are two of the most widely used languages in India. Approximately 530 million people in northern India use the Hindi alphabet, making it the most commonly used Indian script. Bengali ranks as the 2^{nd} most widely spoken language in eastern India. West Bengal, Assam, Tripura, and Manipur states use Bengali scripts for writing [4]. Approximately 100 million individuals in eastern India use the Bengali alphabet for communication. Figure 1 shows samples of our handwritten documents in Hindi and Bengali. Acquiring handwritten datasets is difficult due to the dependence on accessible writers, which might be bothersome. There is a shortage of publicly available

Indian handwritten datasets; here, we have created a dataset AIO-HB(All in one - Hindi-Bengali Dataset), including official Hindi and Bengali Indian scripts. The AIO-HB(All in one - Hindi-Bengali Dataset) has significant text pages, lines, and words/sub-words for each script. These datasets can potentially advance automatic script recognition ability in multi-script texts within the research community. These datasets can be utilised for various applications in Document Image Analysis, including script sentence recognition/understanding, text-line segmentation, word segmentation/recognition, word spotting, handwritten and machine-printed text separation, and writer identification, in addition to script identification.



गङ्गा के किनारे अमरूद, अंगूर और आम के पेड़ों की छाए में बैठे ऊँट के पास खड़े
नेत्रहीन ऐरावत नामक चञ्चल हाथी ने पंकज का फूल उठाकर हौदे में रखे गणेश के
शीर्ष पर डाला और चिंघाड़ उठा, और ईख का गट्ठर सूंड से उठाकर मुँह में
डालकर इत्मीनान से चबाने लगा, जिसे देखकर झांझर, ढोल मंजीरे जोर से बजने
लगे, पटाखों के धमाके हुए और ज्ञानी ऋषियों भक्त गणों ने ओं नमः शिवाय के
गुहार से सारे क्षेत्र को गुंजा दिया और थाली में सिक्के डाल दिए।

মহারাজ উষাকালে বৃহৎ ঐরাবতে রাজপথের অর্দ্ধেক প্রদক্ষিণ করতঃ
হঠাৎ উক্তি করিলেন, "এই ঢোল পাখোয়াজ, আধ্যাত্মিক সঙ্গীতের ঝংকার
ও সৌরভে ক্রন্দিত আড়ম্বরপূর্ণ রঙিন জীবনের অন্তরালে যে বিষণ্ণ
দারিদ্র তাহা কি ঈশ্বর ঔদাসীন্য, অপ্রচলিত বৈদেশিক নীতি নাকি,
বারংবার অহিত পুরানো এবং আত্মঘাতী ডামাডোলের ফসল?"

বর্ষামুখর দিন শেষে, ঊর্দ্ধপানে চেয়ে যখন আষাঢ়ে গল্প শোনাতে
বসে ওসমান ভূঁইঞা, ঈশান কোনে তখন অন্ধকার মেঘের আড়ম্বর,
সবুজে ঋদ্ধ বনভূমির নির্জনতা চিরে থেকে থেকে ঐরাবতের ডাক, মাটির
উপর শুকনো পাতা ঝরে পড়ে ঔদাসীন্যে, এবং তারই ফাঁকে জন্মে
থাকা ঢের। পুরোনো গভীর দুঃখ হঠাৎ যেন বৃষ্টিতে ধুয়ে মুছে ধূসর
জীবনে। রংধনু এনে দেয়।

Fig. 1. a) Hindi Sample image, and b) Bengali Sample image

The present research explores the evolution of Hindi and Bengali scripts, as outlined in Sect. 2. Section 3 provides a thorough analysis that compares the found characteristics in both languages. Section 4 concisely summarises the AIO-HB dataset, including a comparison with conventional surveys, the methodology and criteria used for data collection, and the tools utilised for raw data preparation. This study analyses the methods of feature extraction and classification for handwritten text documents. It also addresses the problems associated with this field, gives potential avenues for future research, and provides a comparative analysis with existing state-of-the-art works in Sect. 5. At last, the paper concludes in Sect. 6.

2 Origin of Hindi and Bengali Scripts

The evolution of the Hindi [14] and Bengali [3] language is an intricate process shaped by historical, cultural, and social influences. Hindi and Bengali, both Indo-European languages, are believed to have originated from Sanskrit, Magadhi Prakrit, and Pali, with ancient Sanskrit serving as the foundation for several contemporary Indian languages. Ancient Prakrits, local languages, gradually transformed into Apabhramsa, the forerunner of contemporary Indo-Aryan languages, shown in Fig. 2 the periodic diagram of language evolution.

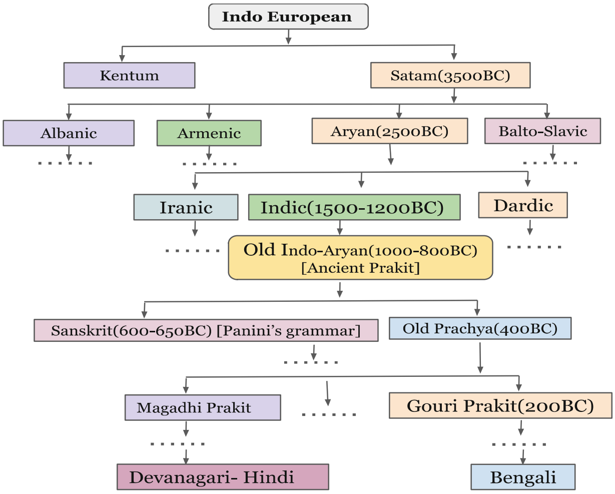


Fig. 2. The evolution of Hindi and Bengali language throughout history

Due to invasions and cultural interactions, Hindi evolved as a unique language throughout the mediaeval era. The Bhakti and Sufi movements profoundly impacted Hindi literature and language throughout mediaeval India. Esteemed poets such as Kabir, Surdas, and Tulsidas contributed to developing devotional poetry in local languages. The advent of European colonisers, namely the British, brought forth novel linguistic impacts, notably an influx of English vocabulary. During the 19^{th} century, there were attempts to establish a standard form of Hindi and encourage its use as a language for literature and administration. This led to modern Hindi grammar, spelling, and vocabulary creation.

The Bengali language has developed into three stages: Old Bengali, Middle Bengali, and Modern Bengali. Old Bengali, originating from 650 A.D., was often used by clergy and intellectuals in Bengali literary compositions. The earliest known literature is Charyapada, a compilation of mystical poetry rooted in Buddhism. In the 14^{th} century, the Sultanate of Bengal established Bengali as the official court language, which later became vernacular. Middle Bengali was affected by Persian imported by the Mughals in the 16^{th} century. Modern Bengali, a dialect spoken in the Nadia area of Bengal, emerged around the Battle of Plassey in 1757. The language is divided into formal and informal forms, known as "$ShuddhoBhasha$" and "$CholitoBhasha$". It primarily incorporates vocabulary from Magadhi Prakrit and Pali and loanwords from many other languages.

Hindi and Bengali writings are formed from Brahmic scripts used in India, Nepal, Tibet, and Southeast Asia. It is derived from the Gupta script, Siddham, and Sharada. Eastern versions of Gupta script were documented in the 7^{th} century CE and eventually supplanted Siddham and Sharada scripts. The Kutila inscription of Bareilly, dating back to 992 CE, displays an early form of Devana-

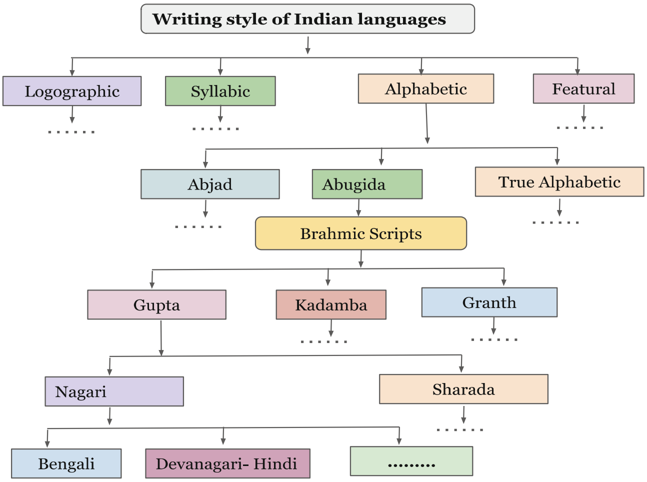


Fig. 3. The evolution of Hindi and Bengali script writing styles

gari. The Sanskrit term Nagari is feminine and signifies "About a town or City," perhaps derived from its association with urban areas shown in Fig. 3. During the mediaeval era, both languages saw substantial changes due to their exchanges with Persian, Arabic, and Turkic languages, which Muslim monarchs influenced. The Persian vocabulary and grammatical structures were incorporated into the language, enhancing its lexicon. Both languages are intricately interwoven with the area's cultural, historical, and linguistic advancements.

3 Characteristics of Bengali and Hindi Scripts

Writing various scripts would result in distinct writing styles, a commonly held assumption; however, it is not universally accurate. The writing style of a screenplay is mostly determined by its structure, which we often refer to as $characteri-stics$. In this part, we are examining methods for determining the distinctions and resemblances between them. Regarding script identification, the differences and similarities may be assessed by examining the characteristics. Features may include elements such as '*shirorekha*' or '*matra*', '*loops*', '*circularity*', '$rectang-ularit$'y, and '*tinycomponents*'. Various script-specific intrinsic traits may be used to differentiate one script from another. Most letters in Hindi and Bengali scripts have a horizontal line, known as '$Matra/Shirorekha$', located at the top section along with a baseline. When two or more letters are arranged horizontally to create a word, the horizontal lines often make contact and produce a continuous line, as shown in Fig. 4.

The Hindi and Bengali scripts consist of 51 fundamental letters, whereas Hindi has 14 vowels and 37 consonants. The contemporary Bengali script has 11 vowels and 40 consonants in its alphabet. Figure 5 displays the fundamental

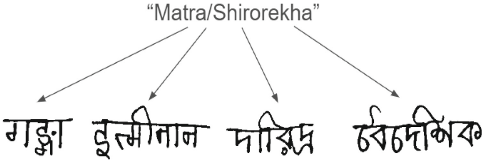
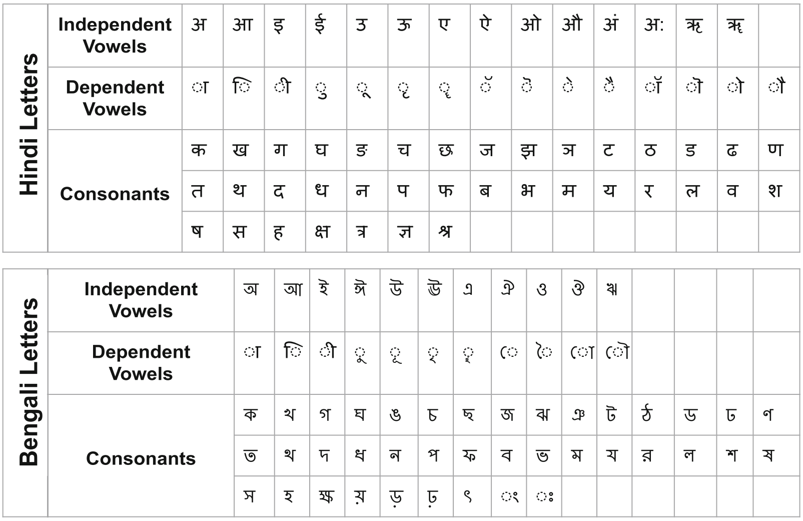


Fig. 4. The Hindi and Bengali words include a horizontal line

letters of the Bengali and Hindi scripts. Indian scripts are written from left to right; however, some pair glyphs may be positioned to the left of their source character for visual representation. Both the Hindi and Bengali scripts use a modified form of a vowel when it follows a consonant. This modified vowel is positioned to the left, right, both left and right, or bottom of the consonant. These altered forms are referred to as modified letters. Due to their topological placement, these modifiers complicate the character segmentation process in Hindi and Bengali scripts. A consonant or vowel that follows a consonant may occasionally take on a compound orthographic form, referred to as a compound character. To get comprehensive information on the Hindi and Bengali scripts, please see reference [2].



Hindi Letters	**Independent Vowels**	अ	आ	इ	ई	उ	ऊ	ए	ऐ	ओ	औ	अं	अ:	ऋ	ॠ	
	Dependent Vowels	◌ा	◌ि	◌ी	◌ु	◌ू	◌ृ	◌ॄ	◌ॅ	◌ॆ	◌े	◌ै	◌ॉ	◌ॊ	◌ो	◌ौ
	Consonants	क	ख	ग	घ	ङ	च	छ	ज	झ	ञ	ट	ठ	ड	ढ	ण
		त	थ	द	ध	न	प	फ	ब	भ	म	य	र	ल	व	श
		ष	स	ह	क्ष	त्र	ज्ञ	श्र								
Bengali Letters	**Independent Vowels**	অ	আ	ই	ঈ	উ	ঊ	এ	ঐ	ও	ঔ	ঋ				
	Dependent Vowels	◌া	◌ি	◌ী	◌ু	◌ূ	◌ৃ	◌ৄ	◌ে	◌ৈ	◌ো	◌ৌ				
	Consonants	ক	খ	গ	ঘ	ঙ	চ	ছ	জ	ঝ	ঞ	ট	ঠ	ড	ঢ	ণ
		ত	থ	দ	ধ	ন	প	ফ	ব	ভ	ম	য	র	ল	শ	ষ
		স	হ	ক্ষ	য়	ড়	ঢ়	ৎ	◌ং	◌ঃ						

Fig. 5. The letters of Hindi and Bengali languages

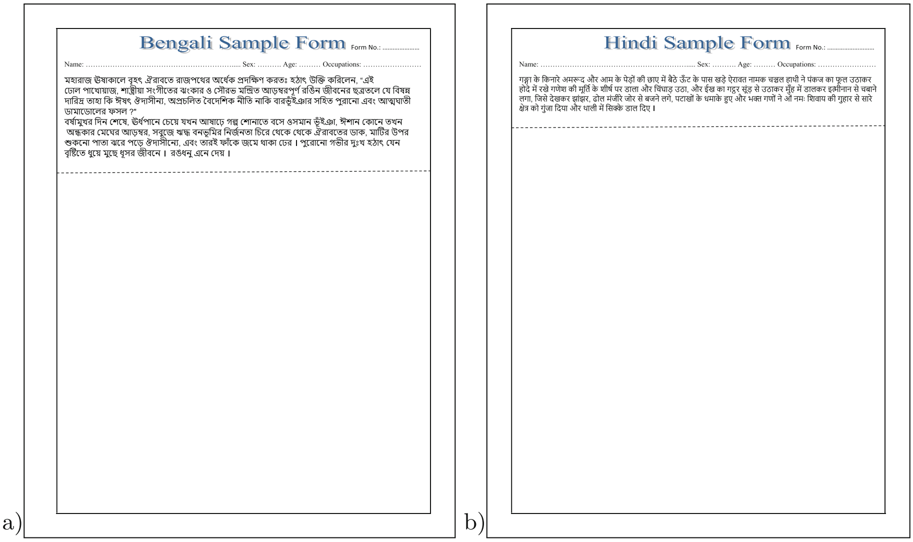
4 A Brief Overview of Our AIO-HB(All in One Hindi Bengali) Dataset

Collecting data is an intensive and monotonous operation in pattern recognition work. Data sample preparation has maintained specific guidelines and methodologies. Sample annotations would be collected so that the specific machine-learning techniques work well. Finally, our dataset is compared with several publicly available benchmark datasets.

4.1 Methodology and Guidelines for Collecting Data

Collecting data gets more difficult when there is a large range of demographic variables. Gathering handwritten data from several authors in different regions is difficult. Handwritten papers are composed in an organised way, such as in pre-formatted forms. The technique was used in our situation as one sheet required volunteers to write the supplied text in the designated place. Each writer received a pre-formatted form. The next paragraph will cover the preparation and norms for creating a data-gathering form.

The first stage included creating a standardised format for sample data collection for both languages. These sample formats were developed at our laboratory, as seen in Fig. 6. The format consists of a header and content. The header field, positioned at the topmost position, includes the writer's name, gender, age, and occupation. To simplify the process, we have included specific information such as '*Male*' or '*Female*' in sex tags and put jobs in occupation tags. This allows the writer to indicate the proper decision easily. The text of the sample paper



a)

Bengali Sample Form Form No.: ....

Name: .... Sex: .... Age: .... Occupations: ....

মহারাজ ঊষাকালে বৃহৎ ঐরাবতে রাজপথের অর্ধেক প্রদক্ষিণ করতঃ হঠাৎ উক্তি করিলেন, “এই ঢোল পাখোয়াজ, শাস্ত্রীয়া সংগীতের ঝংকার ও সৌরভ মন্দ্রিত আড়ম্বরপূর্ণ রঙিন জীবনের ছত্রতলে যে বিষন্ন দারিদ্র তাহা কি ঈষৎ ঔদাসীন্য, অপ্রচলিত বৈদেশিক নীতি নাকি বারভূঁইঞার সহিত পুরানো এবং আত্মঘাতী ডামাডোলের ফসল ?"

বর্ষামুখর দিন শেষে, ঊর্ধপানে চেয়ে যখন আষাঢ়ে গল্প শোনাতে বসে ওসমান ভূঁইঞা, ঈশান কোনে তখন অন্ধকার মেঘের আড়ম্বর, সবুজে ঋদ্ধ বনভূমির নির্জনতা চিরে থেকে থেকে ঐরাবতের ডাক, মাটির উপর শুকনো পাতা ঝরে পড়ে ঔদাসীন্যে, এবং তারই ফাঁকে জমে থাকা ঢের। পুরোনো গভীর দুঃখ হঠাৎ যেন বৃষ্টিতে ধুয়ে মুছে ধূসর জীবনে। রঙধনু এনে দেয়।

b)

Hindi Sample Form Form No.: ....

Name: .... Sex: .... Age: .... Occupations: ....

गङ्गा के किनारे अमरूद और आम के पेड़ों की छाए में बैठे ऊँट के पास खड़े ऐरावत नामक चञ्चल हाथी ने पंकज का फूल उठाकर होदे में रखे गणेश की मूर्ति के शीर्ष पर डाला और चिंघाड़ उठा, और ईख का गट्ठर सूंड़ से उठाकर मुँह में डालकर इत्मीनान से चबाने लगा, जिसे देखकर झांझर, ढोल मंजीरे जोर से बजने लगे, पटाखों के धमाके हुए और भक्त गणों ने ओं नमः शिवाय की गुहार से सारे क्षेत्र को गुंजा दिया और थाली में सिक्के डाल दिए।

Fig. 6. a) Bengali Sample Form image, and b) Hindi Sample Form image

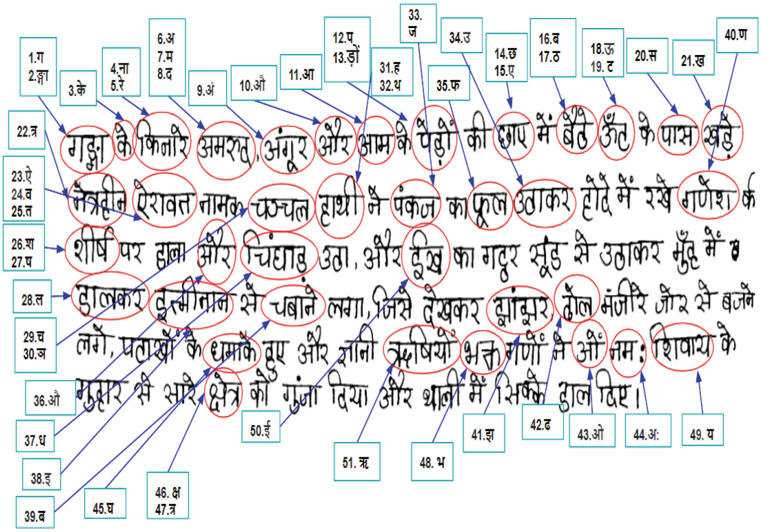


Fig. 7. The page level Hindi sample

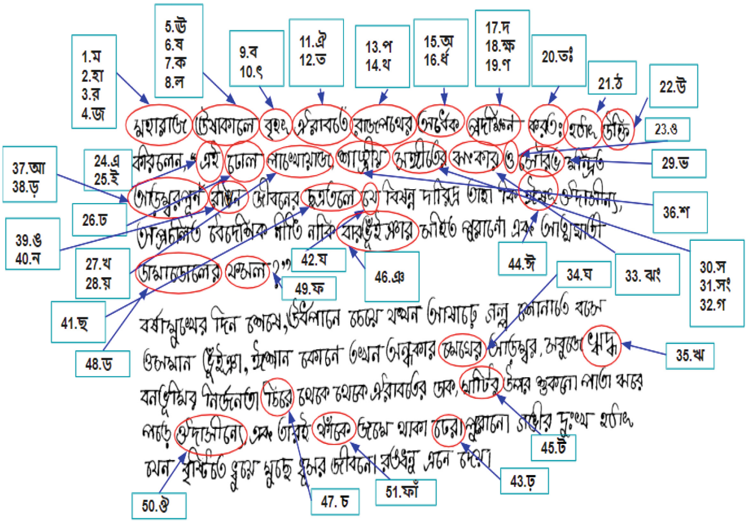


Fig. 8. The page level Bengali sample

was partitioned into two distinct pieces - the higher and lower regions. The higher part had the writer's information on the sample text. The texts' selection was based on consultation with linguistics experts to ensure the inclusion of all letters, including independent and dependent vowels, from both scripts shown in Fig. 7 for the Hindi sample and Fig. 8. for the Bengali sample. For sample collecting, several sources such as news articles, novels, tales, and state board and university curriculum materials were included. This allows us to include a wide range of content and ensure maximum flexibility within the samples. The bottom portion of the sample paper was intentionally left empty, requiring the writer to transcribe the provided material using their own handwriting. The participants were instructed to fill in the empty section of the form with the provided material without any limitations or limits on the choice of pen, ink colour, or writing style. We deliberately tried to gather data from people of varying ages and occupations. In addition, we have gathered data from various locations such as offices, homes, colleges, and schools to guarantee a wide range of writing styles. Most of the scripts, with a few exceptions, were authored by native authors, ensuring they handled over 95% of the instances. Upon receiving the sample sheets, the text sheets were digitised using one HP Laser MFP scanner at 600 dots per inch (dpi) resolution and saved in RGB scales. The final photos were saved to allow users to manipulate them according to their requirements.

Each picture file has been given a name as $<$ *Unique language id* $>$ $<$ 4 *digits unique writer id* $>$ $<$ 2 digit *sample no.* $>$. For example, a Hindi sample file is named '40001_01', where '4' stands for Hindi language id, '0001' stands for Writer id and '01' stands for sample image serial number. As the '*jpg*' file format is selected, the first Hindi sample picture file is saved as '40001_00.jpg'.

Table 1. Age and Gender statistics on AIO-HB (All in one Hindi-Bengali Dataset)

	Hindi			Bengali		
	Male	Female	Total	Male	Female	Total
10–20	12	4	16	84	19	103
21–30	14	8	22	25	7	32
31–40	5	3	8	5	5	10
40–50	4	2	6	6	1	7
Total	35	17	52	120	32	152

4.2 Statistics of Indian Scripts: Hindi and Bengali

Hindi and Bengali are the most widely used scripts in North and Eastern India. We have chosen three distinct categories of textual material for Bengali text pages. This text includes both the fundamental Bengali characters and complex characters. The aforementioned texts were distributed to 150 participants, including diverse ages, genders, and occupations. Statistical information is shown in the Table 1. Ultimately, we successfully gathered a total of 1278 handwritten text pages, which now constitute the Bengali part of AIO-BD dataset. The Bengali dataset consists of about 17k lines of text, totalling 162K words or subwords. A Bengali text page typically comprises around 13.22 lines of text and 127.42 words or subwords. Two examples of pictures of Bengali handwritten text are shown in Fig. 8. The first Bengali text page was stored under the filename

'50001_00.jpg'. The same methodology is used for the Hindi script as preparing the Bengali text. Initially, we generated a paragraph text utilizing various domains. The writings were then sent to a group of 52 persons of different ages, genders, and occupations. The same methodology was used before, accumulating sentences that were ultimately stored in grayscale. The whole quantity of 520 handwritten Hindi text pages was collected, including 4680 lines of text and a total of 51k words/sub-words. On average, each page of Hindi text has 8.89 lines of text and 98.7 words or sub-words. The first Hindi sample sheet was stored under the filename '40001_00.jpg'. Figure 7 displays the example of handwritten text graphics in the Hindi text.

4.3 Synopsis of Our Dataset

The AIO-HB dataset consists of handwritten page-level text pictures comprising many text pages, lines, and words/sub-words in both Hindi and Bengali scripts. Notably, our dataset has a substantial number of authors, namely 202 distinct writers from various regions of India, including diverse age groups, genders, and occupations. The purpose of this dataset is to be used as a standard for identifying handwritten scripts, a growing research challenge in a nation like India that uses several scripts. In addition to various document image analysis applications such as recognising script sentences, segmenting text lines, segmenting words, detecting words, and separating handwritten and machine-printed texts. Furthermore, it can also be used to identify writers across a diverse range of Indian scripts. Our AIO-HB dataset in Table 2 is distinct in its script coverage, larger volume, and huge number of writers and variants, making them distinctive for document analysis.

Table 2. The statistical data of our proposed AIO-HB dataset is compared to other widely accessible datasets at the page and word level

Name of Dataset	Language	No. of Writers	No. of Pages	No. of Lines	No. of Words	Avg. no. of Lines/Page	Avg. no. of Words/Page
CMATERdb1 [27]	Bengali	40	150		30k	21.63	177.3
PBOK [1]	Bengali	199	199	2820	21k	14.17	106.81
PHDIndic_11 [19]	Bengali	42	161	1820	12k	11.3	77.31
	Hindi	60	220	2457	23k	11.16	105.74
RoyDB [24]	Bengali	60			17k		
	Hindi	60			16k		
LAW [12]	Hindi	10			27k		
IIIT-HW-DEV [7]	Hindi	12			95k		
IIIT-INDIC- HW-WORDS [8]	Bengali	24			113k		
AIO-HB [Proposed]	Bengali	150	1278	16906	162k	13.22	127.42
	Hindi	52	520	4680	51k	8.89	98.7

Additionally, we made our AIO-HB dataset into a word-level dataset. Here, we annotated each word image with the following sequences.

Order of Annotation: One digit unique language id & four digits unique writer_id - sample_no - line_no - word_no - x_axis - y_axis - width - height.png shown in the Fig. 9.

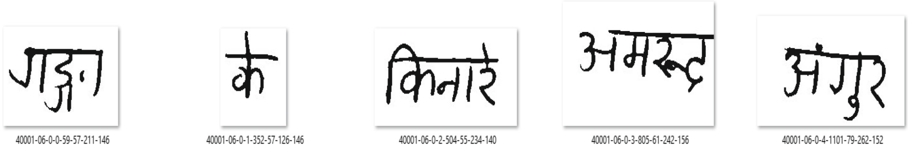


Fig. 9. List of handwritten annotated words

For an annotated handwritten Hindi word Example: 40001-00-0-90-112-352-224.png. Table 3 showing the collection of handwritten annotated words to perform a written identification approach inspired by Sheng He et al. [10] with their CERUG-EN dataset [11]. Sometimes, we attempt to generate synthetic handwritten data [25]; however, this data lacks the precise characteristics of genuine handwriting.

Table 3. The collection of our handwritten text word datasets ready to evaluate with standard Deep learning models

Language	Training DS	Testing DS	Validation DS	Total DS
Hindi	28k	10k	13k	51k
Bengali	91k	30k	41k	162k

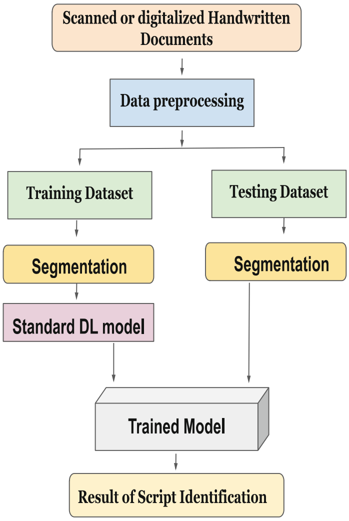


Fig. 10. Flowchart of Handwritten text identification system

5 Analysis of Handwritten Script

Handwritten text identification entails acquiring a scanned picture from an external source for the offline recognition technique. After the first scanning, further pre-processing is carried out on the digital picture to improve its quality and prepare it for segmentation. This pre-processing procedure includes erosion, dilation, opening, closure, and smoothing. The grayscale picture undergoes binarization, which uses the global thresholding approach to transform it into a binary image. The next processes provide the pre-processed picture for segmentation. The process includes fixing holes, expanding the picture, and identifying edges using the Sobel approach.

5.1 Learning Models

Figure 10 illustrates the flowchart of the standard handwritten text identification system. To maintain consistency throughout our dataset, the system resizes collected photos, which may have different sizes, to a standardised scale of 1024 * 1024. The flowchart consists of two sections: training and testing data sets for each language.

5.2 Picture Segmentation

The picture processing procedure entails examining the background and segmenting the handwritten texts using a thresholding technique [26]. This technique distinguishes the main subject from the surrounding elements by transforming the picture into a binary format. The ideal threshold divides pixels based on intensity levels, allowing for the detection of the area of interest [5], here the words/letters. Next, the words divided into segments are separated from any distracting elements in the backdrop. They are then trimmed and saved for further examination and evaluation. This step improves the legibility and perceptibility of texts for sophisticated text identification and feature extraction.

The distinctive characteristics of Bengali and Hindi scripts provide particular challenges and potential for using state-of-the-art deep learning models such as ResNet152, InceptionV3 and Xception in classifying handwritten text.

ResNet152 [9] is an advanced version of the ResNet architecture, consisting of 152 layers, which enables the detection of more complex and detailed characteristics in data. ResNet152 has shown exceptional performance in task categorising images, attaining the most advanced results on our standardised dataset. Due to its significant depth, this model successfully learns hierarchical characteristics, making it well-suited for the task of handwritten text detection. The text identification architecture using ResNet152 comprises building blocks and layers specifically engineered to extract hierarchical characteristics from our handwritten dataset samples.

InceptionV3 [29] , a very efficient architecture of a convolutional neural network, is well-suited for accurately identifying handwritten text. The multi-scale feature representation of this technology enables the detection and analysis of differences in the size, orientation, and style of handwritten texts. Utilising data augmentation methods such as rotation, scaling, and noise may enhance the variety of the training dataset. Text recognition algorithms and sequence modelling are post-processing approaches that may enhance the precision of identifying handwritten text. Nevertheless, it is essential to consider the dataset's precise needs and distinctive features so that our dataset can be easily used to explore this sophisticated CNN architecture.

Xception [6] is a neural network architecture created by François Chollet in 2017. It uses depthwise separable convolutions to capture spatial connections effectively. Its capacity to extract hierarchical elements from handwritten text pictures, such as strokes, forms, and textures, makes it especially valuable for handwritten text recognition tasks. Additionally, Xception may be used with data augmentation methods to enhance the variety of the training dataset. Post-processing methods such as text recognition algorithms or sequence modelling may enhance the precision of identifying handwritten text. Nevertheless, the efficacy of Xception in recognising handwritten text necessitates exploring various topologies, hyperparameters, and training methodologies using our presized dataset.

5.3 Evaluation and Results

The system in issue was implemented using the Python programming language. The dataset was trained using the Resnet152, InceptionV3, and Xception algorithms using the capabilities of the Google Colab platform.

Word error rate (WER) is a metric used to measure the accuracy of a Handwritten Text Recognition (HTR) system in recognizing handwritten text. The metric quantifies the degree of deviation between the recognized text and the ground truth, which is the real handwritten text. WER is computed using the

Table 4. HTR Evaluation Table of the Hindi script using standard Deep Learning approaches

DL Algos	Training		Validation		Testing	
	Accuracy Rate(%)	Word Error Rate(%)	Accuracy Rate (%)	Word Error Rate(%)	Accuracy Rate (%)	Word Error Rate(%)
Resnet152	91.06	8.94	84.21	15.79	89.79	10.21
Inception V3	92.98	7.02	86.78	13.22	93.14	6.86
Xception	88.49	11.51	89.49	10.51	95.45	4.55

Table 5. HTR Evaluation Table of the Bengali script using standard Deep Learning approaches

DL Algos	Training		Validation		Testing	
	Accuracy Rate(%)	Word Error Rate(%)	Accuracy Rate (%)	Word Error Rate(%)	Accuracy Rate (%)	Word Error Rate(%)
Resnet152	85.07	14.93	92.61	7.39	91.47	8.53
Inception V3	88.72	11.28	89.62	10.38	92.89	7.11
Xception	95.62	4.38	94.51	5.49	94.59	5.41

following equation:

$$WER = \frac{Subs + Ins + Del}{Total\, no.\, of\, words\, in\, the\, Document\, Image}$$

where Subs stands for differing word count, Ins stands for additional word count, and Del stands for missing word count from the HTR system's output.

The data table presented the performance metrics of three different algorithms, ResNet152, InceptionV3, and Xception, when applied to a given task. The metrics are supplied during the training, validation & testing stages, specifically comprising the accuracy and error rates shown in Table 4 and Table 5 respectively for both languages.

These data allow for analyzing and comparing the model's performance in Hindi and Bengali HTR. The Bengali HTR model demonstrates marginally better precision, recall, and F1 score compared to the Hindi HTR model, suggesting improved overall performance in recognizing Bengali text shown in Table 6.

Table 6. Analysis of Deep learning Algorithms for Hindi and Bengali HTR.

DL Algos	Precision		Recall		F1-score	
	Hindi	Bengali	Hindi	Bengali	Hindi	Bengali
Resnet152	0.857	0.862	0.909	0.926	0.882	0.893
InceptionV3	0.889	0.917	0.923	0.932	0.906	0.924
Xception	0.888	0.893	0.922	0.929	0.905	0.911

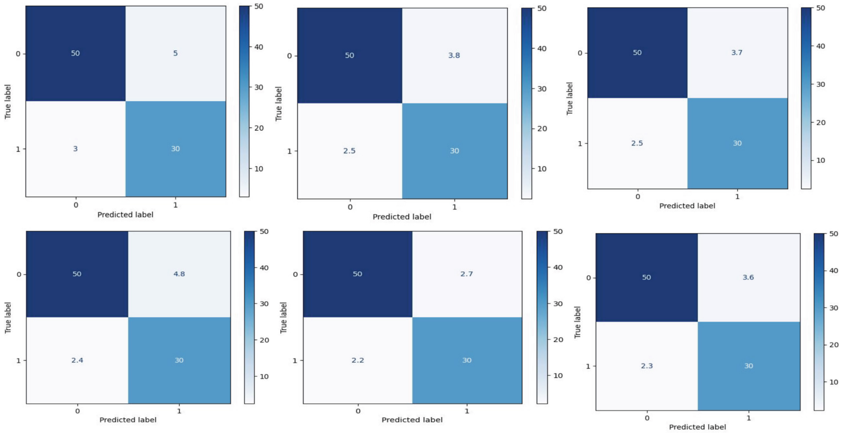


Fig. 11. Confusion matrix: the top three matrices for Hindi HTR utilising ResNet152, InceptionV3, and Xception, respectively, while the bottom matrix represents Bengali HTR.

Based on the evaluation metrics such as precision, recall and F1 score, we generated a confusion matrix where the number of true positives and the number of true negatives are 30 and 50, respectively. Figure 11 presents the confusion matrix of both languages along with three different algorithms, ResNet152, InceptionV3, and Xception.

This analysis facilitates comprehension of the deep learning model's performance in recognising handwritten text across several languages. It can provide information for enhancing or modifying the model's architecture or training procedure. These metrics provide a complete assessment of how well each algorithm learns from the training data and how well they can apply that knowledge to fresh data during the validation and testing phase. The data in the table are used as quantitative benchmarks to evaluate the relative performance of ResNet152, InceptionV3, and Xception in the defined task. Also, a few literature analyses show that script categorization for handwritten documents focuses on various input levels, such as page-level [20], block-level [18], text-line level [17,28], or word-level [13,21].

6 Conclusion

We provide a novel collective handwriting dataset, namely the AIO-HB dataset, which encompasses two prominent spoken Indian languages: Hindi and Bengali. We anticipate that our dataset's extensive size and varied composition, including key Indian languages, will stimulate research efforts to improve and construct resilient systems for recognising handwritten text. Continued enhancement of Indian datasets is crucial for advancing Indian HTR development. Another important objective is to build our handwritten dataset, including annotations at line, paragraph, and page levels. Consequently, future efforts will focus on augmenting the dataset with more authors and incorporating natural variances. We evaluate the performance of a few conventional deep-learning methodologies using our dataset. In the future, we want to expand our efforts to include additional Indian scripts.

References

1. Alaei, A., Pal, U., Nagabhushan, P.: Dataset and ground truth for handwritten text in four different scripts. Int. J. Pattern Recognit. Artif. Intell. **26**(04), 1253001 (2012)
2. Arsenault, P.: Retroflexion in South Asia: typological, genetic, and areal patterns. J. South Asian Lang. Linguist. (JSALL) **4**(1), 1–53 (2017). https://doi.org/10.1515/jsall-2017-0001
3. Banerji, R.D.: The Origin of the Bengali Script. University of Calcutta, Calcutta (1919)
4. Biswas, S., Das, A.K.: Writer identification of Bangla handwritings by radon transform projection profile. In: 2012 10th IAPR International Workshop on Document Analysis Systems, pp. 215–219 (2012). https://doi.org/10.1109/DAS.2012.98

5. Borah, N., Baruah, U., Mahesh, T., Kumar, V.V., Dorai, D.R., Annad, J.R.: Efficient assamese word recognition for societal empowerment: a comparative feature-based analysis. IEEE Access (2023)
6. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1251–1258 (2017)
7. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Offline handwriting recognition on Devanagari using a new benchmark dataset. In: 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), pp. 25–30. IEEE (2018)
8. Gongidi, S., Jawahar, C.V.: IIIT-INDIC-HW-WORDS: a dataset for indic handwritten text recognition. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12824, pp. 444–459. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86337-1_30
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
10. He, S., Schomaker, L.: FragNet: writer identification using deep fragment networks. IEEE Trans. Inf. Forensics Secur. **15**, 3013–3022 (2020)
11. He, S., Wiering, M., Schomaker, L.: Junction detection in handwritten documents and its application to writer identification. Pattern Recognit. **48**(12), 4036–4048 (2015)
12. Jayadevan, R., Kolhe, S.R., Patil, P.M., Pal, U.: Database development and recognition of handwritten Devanagari legal amount words. In: 2011 International Conference on Document Analysis and Recognition, pp. 304–308. IEEE (2011)
13. Kacem, A., Saïdani, A.: A texture-based approach for word script and nature identification. Pattern Anal. Appl. **20**, 1157–1167 (2017)
14. King, C.R.: The nagari pracharini sabha (Society for the Promotion of the Nagari Script and language) of benares, 1893–1914: a study in the social and political history of the Hindi language. The University of Wisconsin-Madison (1974)
15. Lawgali, A., Angelova, M., Bouridane, A.: HACDB: handwritten arabic characters database for automatic character recognition. In: European Workshop on Visual Information Processing (EUVIP), pp. 255–259. IEEE (2013)
16. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. Int. J. Doc. Anal. Recognit. **5**, 39–46 (2002)
17. Obaidullah, S.M., Bose, A., Mukherjee, H., Santosh, K., Das, N., Roy, K.: Extreme learning machine for handwritten Indic script identification in multiscript documents. J. Electron. Imaging **27**(5), 051214–051214 (2018)
18. Obaidullah, S.M., Das, N., Halder, C., Roy, K.: Indic script identification from handwritten document imagesan unconstrained block-level approach. In: 2015 IEEE 2nd International Conference on Recent Trends in Information Systems (ReTIS), pp. 213–218. IEEE (2015)
19. Obaidullah, S.M., Halder, C., Santosh, K., Das, N., Roy, K.: PHDIndic_11: page-level handwritten document image dataset of 11 official Indic scripts for script identification. Multimed. Tools Appl. **77**, 1643–1678 (2018)
20. Obaidullah, S.M., Santosh, K., Halder, C., Das, N., Roy, K.: Automatic Indic script identification from handwritten documents: page, block, line and word-level approach. Int. J. Mach. Learn. Cybern. **10**, 87–106 (2019)
21. Pardeshi, R., Chaudhuri, B., Hangarge, M., Santosh, K.: Automatic handwritten Indian scripts identification. In: 2014 14th International Conference on Frontiers in Handwriting Recognition, pp. 375–380. IEEE (2014)

22. Pratikakis, I., Zagoris, K., Gatos, B., Puigcerver, J., Toselli, A.H., Vidal, E.: Icfhr2016 handwritten keyword spotting competition (h-kws 2016). In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 613–618. IEEE (2016)
23. Rath, T.M., Manmatha, R.: Word spotting for historical documents. IJDAR **9**, 139–152 (2007)
24. Roy, P.P., Bhunia, A.K., Das, A., Dey, P., Pal, U.: Hmm-based Indic handwritten word recognition using zone segmentation. Pattern Recogn. **60**, 1057–1075 (2016)
25. Samanta, P.K., Dutta, A., Biswas, S.: SBGAN: sequential Bengali word image generation model. In: Das, A.K., Nayak, J., Naik, B., Vimal, S., Pelusi, D. (eds.) Computational Intelligence in Pattern Recognition. CIPR 2022. LNNS, vol. 725, pp. 261–271. Springer, Singapore (2023). https://doi.org/10.1007/978-981-99-3734-9_22
26. Sankur, B., Sezgin, M.: Image thresholding techniques: a survey over categories. Pattern Recogn. **34**(2), 1573–1607 (2001)
27. Sarkar, R., Das, N., Basu, S., Kundu, M., Nasipuri, M., Basu, D.K.: Cmaterdb1: a database of unconstrained handwritten Bangla and Bangla-English mixed script document image. Int. J. Doc. Anal. Recognit. (IJDAR) **15**, 71–83 (2012)
28. Singh, P.K., Sarkar, R., Nasipuri, M.: Statistical textural features for text-line level handwritten *Indic* script identification. In: Chaki, R., Saeed, K., Cortesi, A., Chaki, N. (eds.) Advanced Computing and Systems for Security. AISC, vol. 568, pp. 139–151. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-3391-9_9
29. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)

Unconstrained Camera Captured Indic Offline Handwritten Dataset

Ajoy Mondal(✉) and C. V. Jawahar

CVIT, International Institute of Information Technology, Hyderabad, India
{ajoy.mondal,jawahar}@iiit.ac.in

Abstract. This paper presents a diverse compilation of Indic offline handwritten documents. Our dataset comprises 91K handwritten document images captured through unconstrained camera across thirteen Indic languages: *Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Manipuri, Marathi, Oriya, Punjabi, Tamil, Telugu, and Urdu*, contributed by 1,220 writers. This dataset encompasses 2600K words and includes 566,187 unique words featuring diverse content types, such as alphabetic and numeric. Additionally, we establish a high baseline for the proposed dataset, facilitating evaluation, benchmarking and explicitly focusing on word recognition tasks. Our findings indicate that our dataset is an effective training source for enhancing performance on respective datasets. The code, trained model, dataset, and benchmark results are available at https://cvit.iiit.ac.in/usodi/ucciohd.php.

Keywords: Handwritten text recognition · Indic language · Indic script · Camera captured · Unconstrained · Word recognition · Benchmark

1 Introduction

Advancements in Handwritten Text Recognition (HTR) models underscore the necessity for extensive and meticulously annotated handwritten text recognition datasets. These datasets must exhibit diversity in writing and robust annotation to facilitate reliable performance across real-world applications. While English benefits from established datasets like IAM [16,23] and GNHK [14], which are specifically tailored for offline handwritten text recognition, such comprehensive resources are relatively scarce.

Compared to Latin HTR, the exploration of Indic HTR lags due to the scarcity of annotated resources. India's linguistic diversity presents a unique challenge, with numerous languages and scripts in use [1]. Consequently, amassing substantial handwritten datasets across multiple Indic scripts proves to be arduous and costly. Existing annotated datasets [3,7,12,18] for Indic HTR are

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78495-8_21.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 333–348, 2025.
https://doi.org/10.1007/978-3-031-78495-8_21

Table 1. Illustrates comparison of our proposed dataset with existing Indic handwritten text recognition datasets. #W: indicates the number of writers. #Word: indicates the number of words in the dataset. #UW: indicates the number of unique words. IT: indicates imaging type, either flatbed scanner or camera capture.

Script	Language	Dataset	#W	IT	#Word	#UW
Bengali	Assamese	IIIT-INDIC-HW-WORDS	–	–	–	–
		IIIT-Indic-HW-UC	80	camera captured	200K	53049
Bengali	Bengali	PBOK [3]	199	flatbed scanner	21K	925
		ROYDB [18]	60	flatbed scanner	17K	525
		CMATERDB2.1 [7]	300	flatbed scanner	18K	120
		IIIT-INDIC-HW-WORDS	24	flatbed scanner	113K	11295
		IIIT-Indic-HW-UC	158	camera captured	200K	34042
Gujarati	Gujarati	IIIT-INDIC-HW-WORDS	17	flatbed scanner	116K	10963
		IIIT-Indic-HW-UC	47	camera captured	200K	45492
Devanagari	Hindi	ROYDB [18]	60	flatbed scanner	16K	1030
		LAW [12]	10	flatbed scanner	27K	220
		IIIT-INDIC-HW-WORDS	12	flatbed scanner	95K	11030
		IIIT-Indic-HW-UC	118	camera captured	200K	31237
Kannada	Kannada	PBOK [3]	57	flatbed scanner	29K	889
		IIIT-INDIC-HW-WORDS	11	flatbed scanner	103K	11766
		IIIT-Indic-HW-UC	71	camera captured	200K	57474
Malayalam	Malayalam	IIIT-INDIC-HW-WORDS	27	flatbed scanner	116K	13401
		IIIT-Indic-HW-UC	137	camera captured	200K	70230
Bengali	Manipuri	IIIT-INDIC-HW-WORDS	–	–	–	–
		IIIT-Indic-HW-UC	101	camera captured	200K	75531
Devanagari	Marathi	IIIT-INDIC-HW-WORDS	–	–	–	–
		IIIT-Indic-HW-UC	95	camera captured	200K	32038
Oriya	Oriya	PBOK [3]	140	flatbed scanner	27K	1040
		IIIT-INDIC-HW-WORDS	10	flatbed scanner	101K	13314
		IIIT-Indic-HW-UC	75	camera captured	200K	34074
Gurmukhi	Punjabi	IIIT-INDIC-HW-WORDS	22	flatbed scanner	112K	11093
		IIIT-Indic-HW-UC	67	camera captured	200K	18264
Tamil	Tamil	TAMIL-DB [20]	50	flatbed scanner	25K	265
		IIIT-INDIC-HW-WORDS	16	flatbed scanner	103K	13292
		IIIT-Indic-HW-UC	78	camera captured	200K	82052
Telugu	Telugu	IIIT-INDIC-HW-WORDS	16	flatbed scanner	120K	12945
		IIIT-Indic-HW-UC	114	camera captured	200K	39514
Nastaliq	Urdu	CENPARMI-U [19]	51	flatbed scanner	19K	57
		IIIT-INDIC-HW-WORDS	8	flatbed scanner	100K	11936
		IIIT-Indic-HW-UC	79	camera captured	200K	27264

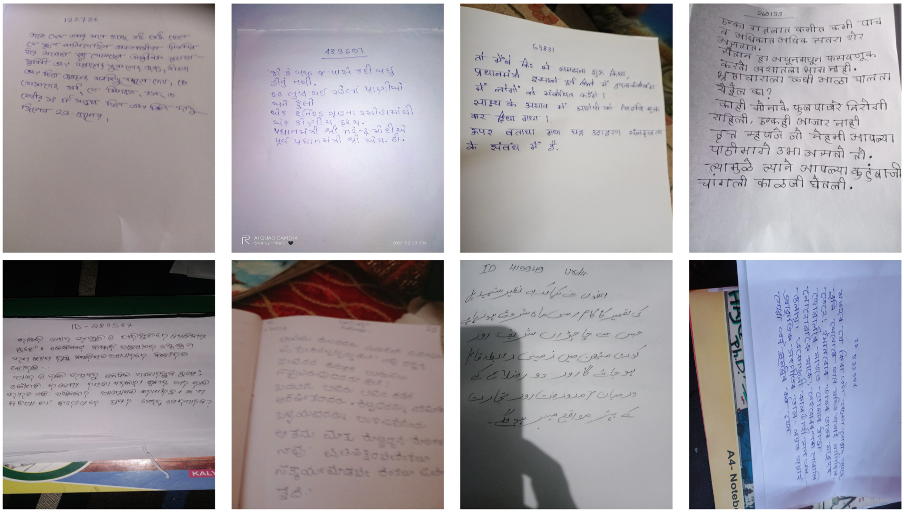


Fig. 1. Shows a few examples of handwritten document images in several Indic languages taken under uncontrolled conditions. This camera- captured images exhibit various characteristics, including blurred text, text with overexposed, perspective text, variation in illumination, unwanted large background, low-resolution text, the text under shadow, oriented text, and others.

limited both in size and breadth. Several initiatives [9–11] have endeavored to narrow the disparity between advancements in Latin and Indian languages by introducing a handwritten dataset spanning ten Indian scripts. However, these datasets typically involve scanned handwritten images captured via flatbed scanners, with writers providing a single word within a box and 20–25 words per page.

With the prevalence of cameras, capturing text in real-world scenarios has become increasingly feasible, allowing us to preserve textual information in pixels. Recently, Zhang *et al.* introduced SCUT-HCCDoc [22], featuring unconstrained camera captured Chinese handwritten documents, while Lee *et al.* [14] presented GNHK, showcasing unconstrained camera captured English handwritten documents. However, such datasets are yet to emerge for offline Indic handwritten documents, highlighting the need for a similar initiative in this domain.

To address this imperative requirement, we present a comprehensive compilation of offline Indic handwritten documents captured in unconstrained settings to facilitate exploration in this domain. Our contributions include the meticulous creation of an innovative dataset explicitly tailored to meet the demands of Indic HTR research. It sets itself apart from existing datasets through a range of key attributes:

- We introduce a dataset, namely *IIIT-Indic-HW-UC*, designed for camera capturing offline Indic handwriting documents in real-world settings (refer Fig. 1). It comprises a wide variety of 91K handwritten documents across

13 languages—*Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Manipuri, Marathi, Oriya, Punjabi, Tamil, Telugu, and Urdu* authored by 1,220 distinct writers all over India and captured by mobile camera. To our knowledge, it is the most extensive and first camera captured dataset for Indic handwritten text recognition (Table 1).

- We offer a baseline model for camera captured Indic handwritten text recognition task (refer Table 3). We employ the cross-dataset analysis method [21] to study generalization aspects comprehensively. It entails training a model on one dataset and assessing its performance on others to gain insights into its adaptability and effectiveness across diverse datasets.

2 Handwritten Datasets in Indic Languages

2.1 Character Level Datasets

Several Indic handwritten character level datasets: DHCD [2], BanglaLekha-Isolated Dataset [8], IITG [5], ISI [6], Kannada-MNIST [17], MNIST-MIX [13] and Urdu [4] are available. The statistics of these datasets are presented in Table 2. These datasets offer a diverse range of handwritten characters from Indic scripts, making them valuable resources for training and evaluating models for character recognition tasks. Researchers and developers can use them to build and test OCR systems, handwriting recognition algorithms, and other applications requiring character level recognition in Indic languages.

Table 2. Illustrates statistics of existing character level handwritten datasets in Indic languages. Script/Lang.: indicates script or language, #Image: indicates the number of images, #W: indicates the number of writers, and #Char. indicates the number of characters.

Script/Lang.	Dataset	#Image	#W	#Char.	Type
Devanagari	DHCD	92,000	–	46	character
Bengali	BanglaLekha-Isolated	166,105	–	84	10 numerals 50 basic characters 24 compound characters
Assamese	IITG	12,000	–	52	characters
Devanagari	ISC	22,556	1049	10	numerals
Bengali		12,938	556	10	numerals
Oriya		5970	356	10	numerals
Kannada	Kannada-MNIST	60,000	65	10	numerals
Urdu	MNIST-MIX	45,000	900	50	characters

2.2 Word Level Datasets

Several datasets containing word level handwritten samples for various Indic languages, such as PBOK [3], ROYDB [18], CMATERDB2.1 [7], CENPARMI-U [19], LAW [12], TAMIL-DB [20], IIIT-HW-DEV [9], IIIT-HW-TELUGU [10], and IIIIT-INDIC-HW-WORDS [11], are publicly available. Table 1 presents the statistics of these datasets. The table reveals that PBOK, ROYDB, CMATERDB2.1, CENPARMI-U, LAW, and TAMIL-DB datasets contain a larger number of writers compared to the *IIIT-INDIC-HW-WORDS* dataset. However, the number of words and unique words in the *IIIT-INDIC-HW-WORDS* dataset significantly exceeds those in the PBOK, ROYDB, CMATERDB2.1, CENPARMI-U, LAW, and TAMIL-DB datasets. Table 1 presents the statistics of existing word level handwritten text recognition datasets alongside our newly created dataset. The table reveals that the existing datasets have limitations such as a small number of word level images, lack of writer variations and writing styles, limited linguistic diversity, and samples collected in constrained environments. Additionally, many of these datasets are not publicly available for research. These limitations make it challenging to generalize OCR models for high accuracy on real and diverse handwritten documents. To address these issues, it is necessary to create a dataset with a larger size, diverse writing styles, samples from unconstrained environments, and greater linguistic diversity.

3 IIIT-Indic-HW-UC Dataset

We create a larger, more diverse dataset of offline handwritten documents captured by cameras, known as the *IIIT-Indic-HW-UC* dataset. Compared to the previous version, this dataset features increased language coverage, word count, writer diversity, writing conditions, imaging processes, and ground truth annotation. It includes thirteen major Indic languages: *Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Manipuri, Marathi, Oriya, Punjabi, Tamil, Telugu, and Urdu*. It consists of 200K word images written by more than 50 writers per language. Writers are required to write corresponding handwritten paragraphs in A4 size white pages, with no constraints on writing style. Handwritten pages are captured using a mobile camera instead of a flatbed scanner. Ground truth annotation is provided at the page level, containing bounding boxes, reading order, textual transcriptions, and the language of all words on the page[1]. We discuss more on it further in the following subsections.

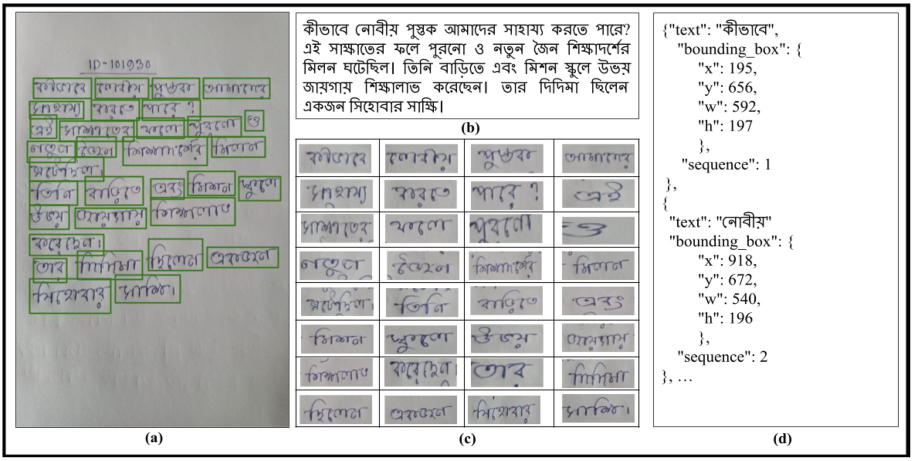
3.1 Data Collection and Annotation

For each language, there are 7K text paragraphs created from the available text corpus[2][3] covering a wide range of topics. Unique id is associated with a para-

[1] However, in this work, we release only individual word level images and their corresponding ground truth transcriptions.

[2] https://ufal.mff.cuni.cz/~majlis/w2c/download.html.

[3] https://corpora.uni-leipzig.de/en?corpusId=ben_wikipedia_2021.



কীভাবে নোবীয় পুস্তক আমাদের সাহায্য করতে পারে? এই সাক্ষাতের ফলে পুরনো ও নতুন জৈন শিক্ষাদর্শের মিলন ঘটেছিল। তিনি বাড়িতে এবং মিশন স্কুলে উভয় জায়গায় শিক্ষালাভ করেছেন। তার দিদিমা ছিলেন একজন সিহোবার সাক্ষি।

```
{"text": "কীভাবে",
  "bounding_box": {
     "x": 195,
     "y": 656,
     "w": 592,
     "h": 197
     },
   "sequence": 1
},
{
 "text": "নোবীয়"
 "bounding_box": {
     "x": 918,
     "y": 672,
     "w": 540,
     "h": 196
     },
   "sequence": 2
}, ...
```

Fig. 2. Illustrates a single Bengali annotated page alongside its standard representation. In (a), a single annotated page from our dataset is depicted. (b) displays the actual text sequence considered as the ground truth. (c) represents all cropped word level images. Lastly, (d) represents textual transcription, bounding box, and sequence of words in a page encapsulated within the JSON file.

graph. Each paragraph contains at most 50 words. Users from any geographical area in India have reading and writing capability for any/all 13 Indic languages to be the authentic writers for the data collection. Any authentic writer can write at least 100 and at most 200 paragraphs in a language selected by the writer. The writer must write one paragraph on one A4 sized page. There is no other constraint on the writing. After writing the paragraph(s), the writer takes a picture of the handwritten page with a mobile camera and shares it with us. There are, on average, 50–100 writers for each language to write 7K text paragraphs. Same paragraphs can be written by multiple writers. With the involvement of 1220 writers, we collect a diverse set of 91K handwritten document images corresponding to 91K text paragraphs; each document image is annotated at the page level. The annotation includes complete text paragraphs and bounding boxes, reading order, textual transcriptions, and the language of all words on the page. A sample annotated handwritten document image is depicted in Fig. 2. Figure 2(a) illustrates the ground truth bounding boxes, Fig. 2(b) shows textual transcription for complete document image, Fig. 2(c) depicts cropped word level images, and finally Fig. 2(d) reveals how the ground truth information (textual transcriptions, bounding boxes, and sequences of words) is stored in JSON format.

3.2 Feature and Statistics

Diversity: The handwritten documents contributed by individuals reflecting various age groups, educational backgrounds, and professional experiences across

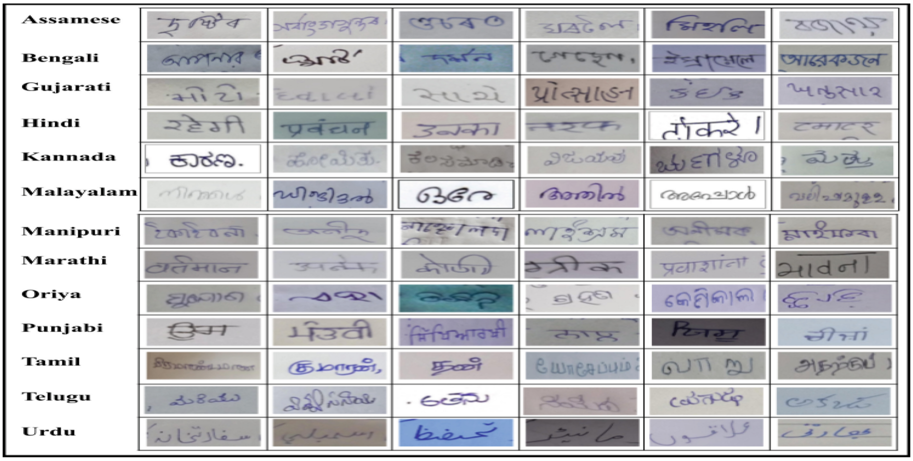


Fig. 3. Show several instances of word level images across various languages, written by multiple writers from our dataset.

India represent a diverse collection. Capturing these handwritten documents using a mobile camera under unconstrained settings presents numerous challenges, including blurred text, text with overexposed, perspective text, variation in illumination, unwanted large background, low-resolution text, the text under shadow, oriented text, and others. Figure 1 illustrates a few sample handwritten document images captured under these unconstrained settings. Furthermore, in Fig. 3, we provide several sample word level images of all thirteen languages written by different writers from our dataset, showcasing a wide range of words and highlighting variations in style, image quality, and other aspects.

Since documents are written by various writers all over India, there is enough diversity among documents written by two different writers. Figure 4 shows a few sample word level images of Hindi written by two different writers: Writer-1 and Writer-21. It highlights that there is still enough variation in writing style and imaging quality between the two writers. Since one writer can write at least 100 and at most 200 pages, for a writer, among document images, there are also enough variations in style and imaging quality because of camera capture. Figure 5 shows a few sample word level images of Bengali and Hindi languages written by the same writer.

Document Image Resolution Distribution: Writers employ their smartphone cameras to photograph handwritten documents, resulting in variations in the resolution of the captured images. Acknowledging that high-resolution document images offer clear text content, facilitate effective model training, and yield superior performance during testing is crucial. Incorporating document images with diverse resolutions ranging from 1600 × 720 to 4608 × 3456 introduces variability in content visibility, thereby enhancing the robustness of the model.

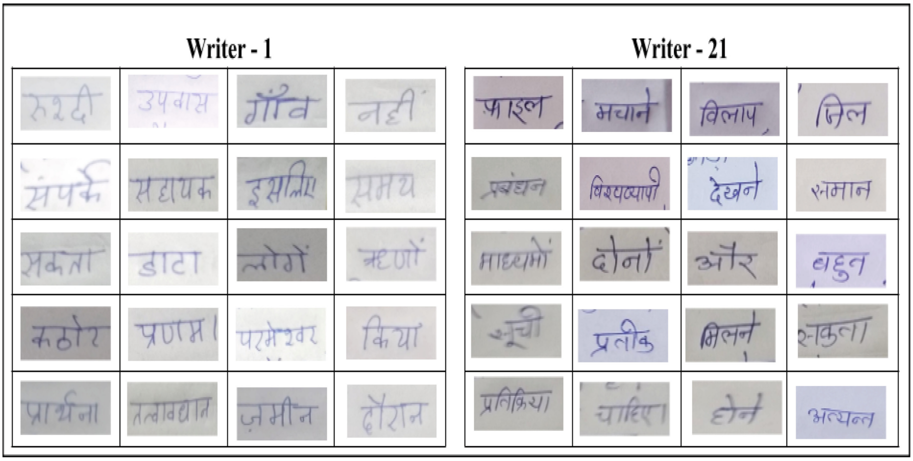


Fig. 4. Presents explicitly sample word level images from just two different writers, namely, writer-1 and writer-21.

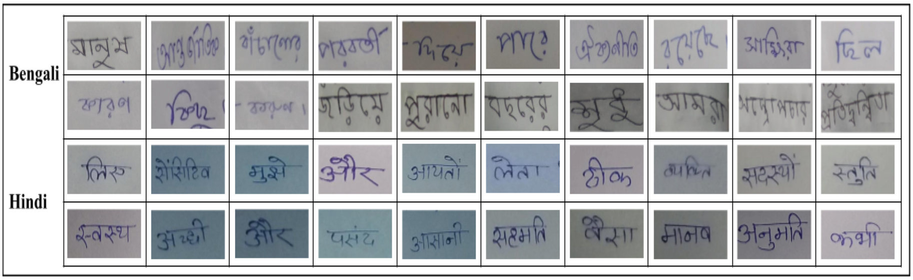
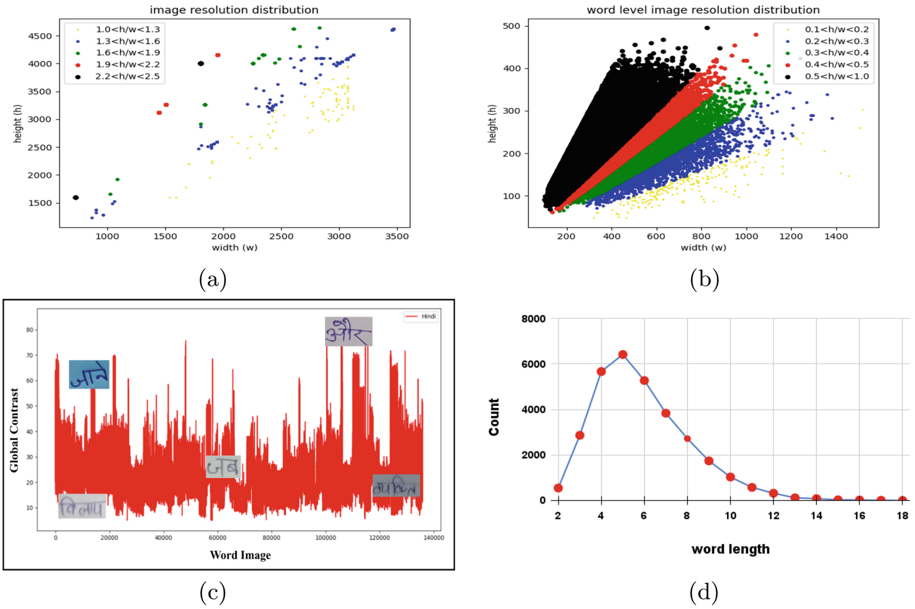


Fig. 5. Show word level images from different document images written by the same writer for Bengali and Hindi languages.

Figure 6(a) highlights the distribution of resolution of handwritten document images by different writers for the Hindi language in our dataset.

Word Level Image Resolution Distribution: Variations in text content and individual writers contribute to differences in the resolution of handwritten word level images. This diversity in word level image resolution improves the model's generalization ability. As depicted in Fig. 6(b), the distribution of resolutions in Hindi word level images highlights the dataset's variability. Most word level images have a height-to-width ratio between 0.5 and 1.0, so the dataset encompasses word level images of varying resolutions. Including word level images with diverse resolutions enriches the dataset, enabling the model to accommodate a broader range of visual attributes and enhance its performance across various writing styles and conditions.



(a) (b) (c) (d)

Fig. 6. Shows (a) document image resolution distribution, (b) word level image resolution distribution, (c) varying global contrast among word images, and (d) distribution of word length for the Hindi language in our dataset.

Contrast of Word Level Images: Word level images are extracted from handwritten document images captured by various mobile cameras, resulting in significant variations in intensities and contrast among the images. To evaluate the recognition ease of each word level image, we adopt the global contrast strategy [15]. Figure 6(c) illustrates the diverse global contrast levels observed among word level images in Hindi. The figures illustrate contrast levels ranging between 10 and 80. This variability in intensity within word level images adds complexity to the dataset, contributing to the development of robust HTR models. However, comprehending and utilizing these variations can enhance the adaptability and effectiveness of HTR algorithms across diverse linguistic contexts.

Text Distribution: We compile a dataset of 7K document images for each language, totaling 91K. Additionally, there are 200K word level images per language, resulting in 2600K word level images encompassing unique 566,187 alphabetic and numeric words. Table 1 presents the unique words for each language in our dataset, denoted by the 7^{th} column. For the Hindi language, we include a plot in Fig. 7 that shows the occurrence of unique words in the dataset. The x-axis shows unique words, and the y-axis shows the occurrence of a particular word on a logarithmic scale. This plot demonstrates a long-tail distribution, confirming the diversity of the dataset[4].

Writer Characteristics: Across India, 1,220 individuals have actively contributed to curating handwritten document images, resulting in a diverse dataset encompassing various handwriting styles, camera specifications, scanning methods, and more. Among these contributors, 70% (854 individuals) are female, while the remaining 30% (366 individuals) are male. Within the male cohort, 23 individuals are identified as left-handed, with 343 being right-handed. Among the female contributors, 34 individuals are left-handed, while the majority, specifically 820, are right-handed. Notably, a significant portion of the contributors falls within the age range between 20 to 40.

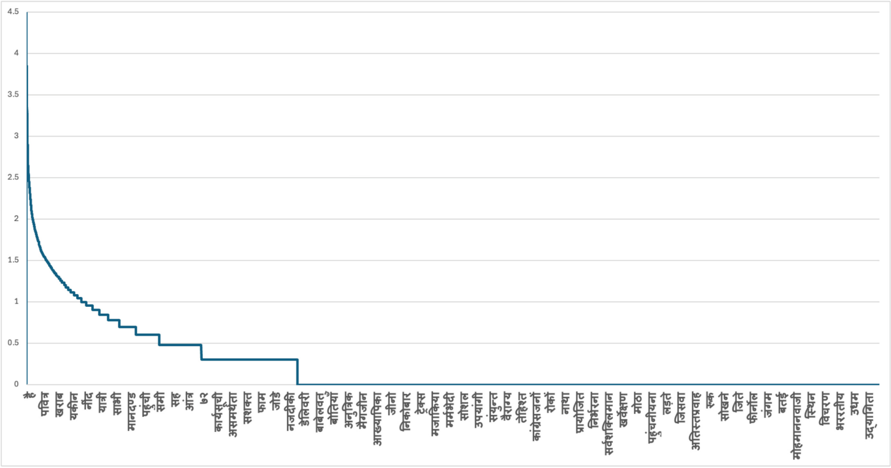


Fig. 7. Shows the distribution of unique Hindi words in the dataset.

Dataset Splits: To furnish an extensive training dataset for deep learning models, our dataset, consisting of 2600K word level images, has been partitioned into 1950K word level images for training, 260K word level images for validation, and 390K word level images for testing. For each language, the dataset includes 200K word level images, among them 75% (i.e., 150K), 10% (i.e., 20K), and 15% (i.e., 30K) word images for training, validation, and test sets, respectively.

Comparison with Existing Datasets: Table 1 comprehensively compares our proposed dataset and existing offline Indic handwritten text recognition datasets, highlighting significant disparities and advantages. Various factors, such as dataset size, diversity in handwriting styles, and the inclusion of diverse texts, are

[4] Plots of the occurrence of unique words for other languages in the supplementary material.

meticulously examined. Our *IIIT-Indic-HW-UC* dataset is twice as large as *IIIT-INDIC-HW-WORDS* regarding the number of word level images, resulting in a more extensive collection of handwritten document images. In contrast to existing datasets where images are scanned using flatbed scanners, our dataset comprises images captured using mobile cameras under unconstrained environments. Camera capture introduces various challenges such as blurred text, overexposed text, perspective distortion, variations in illumination, extensive unwanted backgrounds, low-resolution text, text under shadow, and oriented text, among others, in handwritten document images. Across all languages in our dataset, the number of writers exceeds that of *IIIT-INDIC-HW-WORDS*, indicating a more diverse range of writing styles. For instance, in the Bengali language, the PBOK and CMATERDB2.1 datasets boast more writers (199 and 300, respectively) compared to our dataset, with 158 writers. However, our dataset contains a more significant number of unique words (34042) than the PBOK and CMATERDB2.1 datasets, which have 925 and 25 unique words, respectively. Similarly, in the Oriya language, while the PBOK dataset has more writers (140) than our dataset (75), our dataset surpasses PBOK in terms of the number of unique words (34074 versus 1040). These distinguishing characteristics render our dataset larger, covering major Indic languages and offering diverse word level images compared to existing datasets.

4 Benchmark Experiments

4.1 Experimental Settings

Baseline: We utilize the network architecture proposed by Gongidi *et al.* [11], depicted in Fig. 8, as the baseline for our experiment. This network consists of four main modules: the Transformation Network (TN), Feature Extractor (FE), Sequence Modeling (SM), and Predictive Modeling (PM). The Transformation Network comprises six plain convolutional layers with 16, 32, 64, 128, 128, and 128 channels, each followed by a max-pooling layer of size 2×2 and a stride of 2. The Feature Extractor module adopts the ResNet architecture, while the Sequence Modeling module employs a 2-layer Bidirectional LSTM (BLSTM) with 256 hidden neurons in each layer. Finally, the Predictive Modeling module utilizes Connectionist Temporal Classification (CTC) for character decoding and recognition by aligning the feature sequence with the target character sequence. Further details can be found in [11].

Implementation Details: The baseline model is trained using a single NVIDIA GeForce GTX 1080 Ti GPU. Word level images are resized to dimensions of 96×256 during pre-processing. Stochastic Gradient Descent (SGD) is utilized with the Adadelta optimizer, employing a learning rate of 0.001, a batch size of 64, and a fixed momentum of 0.09. Upon acceptance of the paper, both the trained model and dataset will be released to the public.

Training/Testing Details: The baseline model is trained on the training sets of our *IIIT-Indic-HW-UC*, *IIIT-INDIC-HW-WORDS*, and a combination of

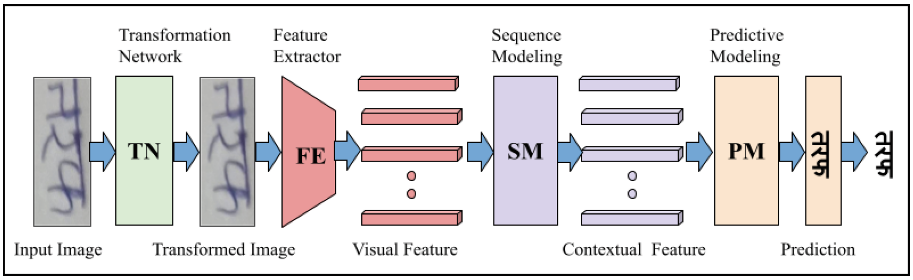


Fig. 8. Shows text recognition through the baseline pipeline.

these two datasets. Following training, we evaluate the performance of the baseline model on the respective test sets of *IIIT-Indic-HW-UC* and *IIIT-INDIC-HW-WORDS* datasets.

Evaluation Metrics: We utilize two widely recognized evaluation metrics, namely, Character Recognition Rate (CRR) (alternatively Character Error Rate, CER) and Word Recognition Rate (WRR) (alternatively Word Error Rate, WER), to evaluate the performance of the baseline. Error Rate (ER) is defined as:

$$ER = (S + D + I)/N, \tag{1}$$

where S represents the number of substitutions, D denotes the number of deletions, I signifies the number of insertions, and N indicates the total number of instances in reference text. In the context of CER, Eq. (1) operates at the character level, and while of WER, Eq. (1) operates at the word level. The Recognition Rate (RR) is defined as:

$$RR = 1 - ER. \tag{2}$$

For CRR, Eq. (2) operates at the character level, and for WRR, it functions at the word level.

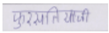
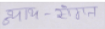
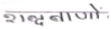
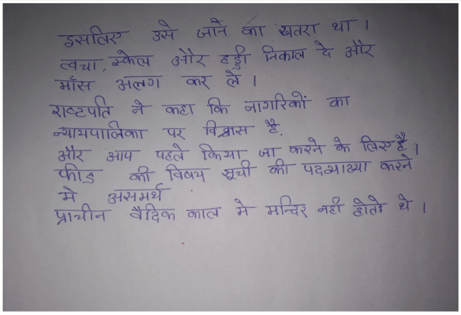
Language	Trained on	Tested on			
		IIIT-INDIC-HW-WORDS		IIIT-Indic-HW-UC	
					
Hindi	IIIT-INDIC-HW-WORDS	फुरसतियाजी फुरसतियाजी	न्याय-संगत न्याय-संगन	लोकतंत्र लोकतंत्र	शब्दबाणों शन्दबाणों
	IIIT-Indic-HW-UC	फुरसतियाजी फुरसतियाजी	न्याय-संगत आययसंगत	लोकतंत्र लोकतंत्र	शब्दबाणों शब्दबाणों
	IIIT-INDIC-HW-WORDS and IIIT-Indic-HW-UC	फुरसतियाजी फुरसतियाजी	न्याय-संगत न्याय-रांगत	लोकतंत्र लोकतंत्र	शब्दबाणों शब्दबाणों

Fig. 9. Visual results obtained by baseline for Hindi language. Ground truth text is highlighted in Blue color. Correctly recognized text is highlighted in Black color. Wrongly recognized text is highlighted in Red color. (Color figure online)

Table 3. Quantitative results on different Indic handwritten datasets. Bold value indicates the best results.

Language	Training Set	Test Dataset			
		IIIT-INDIC-HW-WORDS		*IIIT-Indic-HW-UC*	
		CRR	WRR	CRR	WRR
Assamese	*IIIT-Indic-HW-UC*	–	–	90.98	85.15
Bengali	*IIIT-INDIC-HW-WORDS*	95.72	84.29	79.66	51.51
	IIIT-Indic-HW-UC	85.99	53.82	96.60	89.10
	Both	**96.28**	**84.58**	**96.69**	**89.27**
Gujarati	*IIIT-INDIC-HW-WORDS*	96.16	81.41	81.64	54.14
	IIIT-Indic-HW-UC	85.41	56.96	97.64	88.24
	Both	**97.39**	**87.53**	**98.01**	**88.74**
Hindi	*IIIT-INDIC-HW-WORDS*	96.08	83.09	72.27	40.81
	IIIT-Indic-HW-UC	89.66	63.65	93.88	84.69
	Both	**96.64**	**85.20**	**93.96**	**85.06**
Kannada	*IIIT-INDIC-HW-WORDS*	98.69	92.36	84.52	55.36
	IIIT-Indic-HW-UC	87.96	61.50	97.90	91.49
	Both	**98.83**	**92.41**	**98.15**	**91.59**
Malayalam	*IIIT-INDIC-HW-WORDS*	98.01	89.78	74.97	32.07
	IIIT-Indic-HW-UC	86.77	55.37	96.13	86.74
	Both	**98.16**	**89.81**	**97.05**	**87.07**
Manipuri	*IIIT-Indic-HW-UC*	-	-	90.99	83.38
Marathi	*IIIT-Indic-HW-UC*	-	-	96.32	86.67
Oriya	*IIIT-INDIC-HW-WORDS*	96.02	80.82	84.72	52.35
	IIIT-Indic-HW-UC	85.64	53.13	94.09	80.43
	Both	**96.47**	**82.91**	**94.89**	**81.34**
Punjabi	*IIIT-INDIC-HW-WORDS*	94.87	81.63	69.66	46.60
	IIIT-Indic-HW-UC	83.32	50.11	94.82	87.42
	Both	**96.59**	**86.63**	**94.95**	**87.54**
Tamil	*IIIT-INDIC-HW-WORDS*	98.42	92.19	78.27	41.24
	IIIT-Indic-HW-UC	94.65	73.55	95.31	83.93
	Both	**98.71**	**92.36**	**96.29**	**84.06**
Telugu	*IIIT-INDIC-HW-WORDS*	95.42	76.02	76.73	35.05
	IIIT-Indic-HW-UC	94.50	71.46	93.43	74.34
	Both	**97.50**	**83.60**	**93.68**	**74.84**
Urdu	*IIIT-INDIC-HW-WORDS*	93.50	75.89	71.25	43.94
	IIIT-Indic-HW-UC	75.47	47.07	93.21	82.18
	Both	**96.11**	**85.01**	**94.59**	**82.93**



इसलिए उसे जाने का खतरा था ।
त्वचा, स्केल और हट्टी निकाल दे और
माँस अलग कर ले ।
राष्ट्रपति ने कहा कि नागरिकों का
न्यायपालिका पर विश्वास है
और आप पहले किया जा करने के लिए है।
फीड की विषय सूची की पदव्याख्या करने
मे असमर्थ
प्राचीन वैदिक काल मे मन्दिर नही होतो थे ।

इसलिए उसे जाने का खतरा था ।
त्वचा, स्केल और हट्टी निकात दे और
माँस अलग कर ले ।
राष्ट्पति ने कहा कि नागरिको का
न्यायपालिका पर विश्वास है
और आप पहले किया जा करने के लिए है।
फीड की विषय सूची की पदमख्या करने
मे असमर्थ
प्राचीन वैदिक काल मे मन्दिर नही होतो थे ।

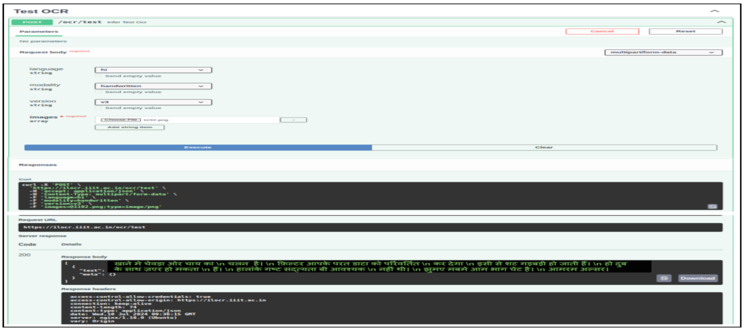
Fig. 10. Page level visual results obtained by baseline for Hindi language. Ground truth text is highlighted in Blue color. Correctly recognized text is highlighted in Black color. Wrongly recognized text is highlighted in Red color. (Color figure online)

4.2 Benchmark Results on Word Level Text Recognition

The performance evaluation results of our baseline model on offline Indic handwritten datasets are presented in Table 3. The table illustrates that when the model is trained on *IIIT-INDIC-HW-WORDS* and tested on both *IIIT-INDIC-HW-WORDS* and *IIIT-Indic-HW-UC*, it achieves notably higher accuracy on *IIIT-INDIC-HW-WORDS* compared to *IIIT-Indic-HW-UC* due to the domain gap (flatbed vs. camera captured and constrained vs. unconstrained) between these datasets. Similarly, when the model is trained on *IIIT-Indic-HW-UC* and tested on both *IIIT-INDIC-HW-WORDS* and *IIIT-Indic-HW-UC*, it achieves better results on *IIIT-Indic-HW-UC* than *IIIT-INDIC-HW-WORDS*, again due to the domain gap between the datasets. We also noticed that when the model is trained on both *IIIT-INDIC-HW-WORDS* and *IIIT-Indic-HW-UC* and then tested on these two datasets, it achieved the highest performance (indicated by bold values in Table 3) for all languages across both datasets. We noticed that when the model was trained with *IIIT-Indic-HW-UC* and tested on *IIIT-INDIC-HW-WORDS*, it achieved better WRR and CRR compared to when the model was trained with *IIIT-INDIC-HW-WORDS* and tested on *IIIT-Indic-HW-UC*. The unique properties of the *IIIT-Indic-HW-UC* dataset, such as unconstrained and camera captured images, contribute to the model's generality, enabling better performance even on the *IIIT-INDIC-HW-WORDS* dataset, which comprises images captured by a flatbed scanner in a constrained environment. This suggests that the *IIIT-Indic-HW-UC* dataset is more diverse and makes the model more generic than the *IIIT-INDIC-HW-WORDS* dataset. Figures 9 and 10 display visual results for the Hindi language at both word and page levels, respectively. In these figures, ground truth text is highlighted in blue, correctly recognized text in black, and wrongly recognized text in red.

4.3 APIs and Web-Based Applications

We develop APIs for handwritten page recognition models across 13 languages and create a web-based application that integrates these APIs to digitize handwritten documents in Indic languages. Figure 11 illustrates the steps for using our web-based APIs to digitize Indic handwritten documents. Users can upload a handwritten document image, select the language, choose the OCR model version and layout version, and then execute the process to obtain the OCR output.



language string: hi
modality string: handwritten
version string: v3
images * required array: 1192.png

Curl

```
curl -X 'POST' \
  'https://ilocr.iiit.ac.in/ocr/test' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'language=hi' \
  -F 'modality=handwritten' \
  -F 'version=v3' \
  -F 'images=@1192.png;type=image/png'
```

Request URL

```
https://ilocr.iiit.ac.in/ocr/test
```

Server response

Code 200

Details

Response body

```
[
  {
    "text": खाने मे चेवड़ा ओर चाय का \n चलन  है। \n फ़िल्टर आपके परत डाटा को परिवर्तित \n कर देगा \n इसी से शट गड़बड़ी हो जाती हैं। \n हो दुब के साथ ज़एर हो मकता \n हैं। \n हालांके गष्ट सद्त्यता बी आवश्यक \n नहीं थी। \n झुमए मबमे आम भाग चेट है। \n आमरम अल्सर।
    "meta": {}
  }
]
```

Response headers

```
access-control-allow-credentials: true
access-control-allow-origin: https://ilocr.iiit.ac.in
connection: keep-alive
content-length: 74
content-type: application/json
date: Wed,10 Jul 2024 09:36:15 GMT
server: nginx/1.18.0 (Ubuntu)
vary: Origin
```

Fig. 11. Shows screen shot of our web-based APIs to digitize Indic handwritten documents.

5 Conclusion

We have introduced a large and diverse dataset called *IIIT-Indic-HW-UC* for Indic offline handwritten text recognition. This dataset contains camera captured images of handwritten documents in thirteen Indic languages: *Assamese, Bengali, Gujarati, Hindi, Kannada, Malayalam, Manipuri, Marathi, Oriya, Punjabi, Tamil, Telugu, and Urdu.* We gathered these samples from various regions in India. The dataset includes 91K text paragraphs written by 1,220 writers, offering a wide range of content. We identified 26K instances of words within these images, including alphabetic and numeric text. Among these, 566,187 instances are unique. Our paper presents benchmark results for text recognition using established architectures, showing that training models with our dataset improves performance on existing offline handwritten datasets. We suggest that future research could explore end-to-end approaches integrating localization and recognition. We invite contributions from researchers and developers to explore new models using our dataset.

Acknowledgment. This work is supported by MeitY, Government of India, through the NLTM-Bhashini project.

References

1. Census 2011. https://censusindia.gov.in/2011-Common/CensusData2011.Html
2. Acharya, S., Pant, A.K., Gyawali, P.K.: Deep learning based large scale handwritten Devanagari character recognition. In: SKIMA (2015)
3. Alaei, A., Pal, U., Nagabhushan, P.: Dataset and ground truth for handwritten text in four different scripts. IJPRAI **26**(04), 1253001 (2012)
4. Ali, H., Ullah, A., Iqbal, T., Khattak, S.: Pioneer dataset and automatic recognition of Urdu handwritten characters using a deep autoencoder and convolutional neural network. SN Appl. Sci. **2**, 1–12 (2020)
5. Baruah, U., Hazarika, S.M.: A dataset of online handwritten Assamese characters. J. Inf. Process. Syst. **11**(3), 325–341 (2015)
6. Bhattacharya, U., Chaudhuri, B.: Databases for research on recognition of handwritten characters of Indian scripts. In: ICDAR (2005)
7. Bhowmik, S., Malakar, S., Sarkar, R., Basu, S., Kundu, M., Nasipuri, M.: Off-line Bangla handwritten word recognition: a holistic approach. Neural Comput. Appl. **31**, 5783–5798 (2019)
8. Biswas, M., et al.: Banglalekha-isolated: a comprehensive Bangla handwritten character dataset. arXiv preprint arXiv:1703.10661 (2017)
9. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Offline handwriting recognition on Devanagari using a new benchmark dataset. In: DAS (2018)
10. Dutta, K., Krishnan, P., Mathew, M., Jawahar, C.: Towards spotting and recognition of handwritten words in Indic scripts. In: ICFHR (2018)
11. Gongidi, S., Jawahar, C.: IIIT-INDIC-HW-Words: a dataset for Indic handwritten text recognition. In: ICDAR (2021)
12. Jayadevan, R., Kolhe, S.R., Patil, P.M., Pal, U.: Database development and recognition of handwritten Devanagari legal amount words. In: ICDAR (2011)
13. Jiang, W.: MNIST-MIX: a multi-language handwritten digit recognition dataset. IOP SciNotes **1**(2), 025002–025010 (2020)
14. Lee, A.W., Chung, J., Lee, M.: GNHK: a dataset for English handwriting in the wild. In: ICDAR (2021)
15. Li, Y., Hou, X., Koch, C., Rehg, J.M., Yuille, A.L.: The secrets of salient object segmentation. In: CVPR (2014)
16. Marti, U.V., Bunke, H.: The IAM-database: an English sentence database for offline handwriting recognition. IJDAR **5**, 39–46 (2002)
17. Prabhu, V.U.: Kannada-MNIST: a new handwritten digits dataset for the Kannada language. arXiv preprint arXiv:1908.01242 (2019)
18. Roy, P.P., Bhunia, A.K., Das, A., Dey, P., Pal, U.: Hmm-based Indic handwritten word recognition using zone segmentation. Pattern Recogn. **60**, 1057–1075 (2016)
19. Sagheer, M.W., He, C.L., Nobile, N., Suen, C.Y.: A new large Urdu database for off-line handwriting recognition. In: ICIP (2009)
20. Thadchanamoorthy, S., Kodikara, N., Premaretne, H., Pal, U., Kimura, F.: Tamil handwritten city name database development and recognition for postal automation. In: ICDAR (2013)
21. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR (2011)
22. Zhang, H., Liang, L., Jin, L.: SCUT-HCCDoc: a new benchmark dataset of handwritten Chinese text in unconstrained camera-captured documents. Pattern Recogn. **108**, 107559 (2020)
23. Zimmermann, M., Bunke, H.: Automatic segmentation of the IAM off-line database for handwritten English text. In: ICPR, vol. 4, pp. 35–39 (2002)

FTC: A Novel Triplet Classification Model for Joint Entity and Relation Extraction

Bin Zhang[1,2], Chaofan Zou[1(✉)], and Wenwen Song[2]

[1] School of Cyber Security and Computer, Hebei University, Baoding, China
zb@hbu.edu.cn, zcf@stumail.hbu.edu.cn

[2] Information Technology Center, Hebei University, Baoding, China
songww@hbu.edu.cn

Abstract. Joint extraction of entities and relations from unstructured text is a crucial task in the construction of knowledge graph and natural language processing. Current methods simplify the implementation by breaking down joint extraction into modular stages. Despite their substantial performance, limitations such as cascading errors and redundancy in relational predictions persist. These issues arise from the oversight of the interdependence and inseparability of the three elements in a triple. In order to tackle these challenges, we introduce a novel Fine-Grained Triplet Classification Model(FTC) that relies on a score-based classifier and a tagging strategy for relation-specific upper triangular. The classifier evaluates if a token pair and relation form a factual triple, while the tagging strategy ensures straightforward and efficient decoding. In comprehensive experiments on four well-established datasets for relation triple extraction, FTC outperforms state-of-the-art baselines, consistently achieving performance improvements across diverse overlapping patterns and complex scenarios involving multiple triples. Our approach acknowledges the interconnected nature of entities and relations, mitigating cascading errors and reducing redundancy in predictions. Through this innovative perspective, the FTC demonstrates its effectiveness in enhancing the accuracy and efficiency of joint entity and relation extraction from unstructured text.

Keywords: Joint Relation Extraction · Triple Overlap Problem · Error Propagation

1 Introduction

Identifying triples (subject, relation, object) and their relations from unstructured text is crucial for knowledge graph construction and natural language processing. Some previous works have proposed using a pipeline approach to address this task, comprising two steps: named entity recognition(NER) [16,22] and relation extraction(RE) [1,15,25,31]. First, entities and their types are identified

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 349–364, 2025.
https://doi.org/10.1007/978-3-031-78495-8_22

from the input text, followed by determining the relation between each entity pair. Although flexible, the pipeline approach ignores interactions between these subtasks and is prone to error propagation [10]. As a result, recent research has focused on developing joint models that capture entities and their relations using a unified framework.

To simplify complex tasks, existing research often segments joint extraction into multiple fundamental modules or phases [28,37]. This approach is categorized into two methods: joint decoding [39] and parameter sharing [7,26,30]. The former integrates entity and relation labels, utilizing sequence labels for the simultaneous identification of both entities and relations. The latter employs a multi-task learning approach, enabling NER and RE models to share encoding layers and optimize their loss functions together. Traditional joint methods [10,14,29] necessitate feature engineering and exhibit lower efficiency. However, with deep learning advancements, several studies [23,26,28,38,39] leveraging deep neural networks have produced promising outcomes.

Redundant relation computation and limited generalization in entity recognition are challenges faced by existing models. Most models predict triples for each relation, resulting in significant computational overhead. Additionally, Overlapping patterns such as Single Entity Overlap (SEO), Entity Pair Overlap (EPO), and Subject Object Overlap (SOO) further complicate the process. Span-based recognition methods, which indicate only the start and end positions of entities, often fail to handle single-character or overlap-based entities. Although some models use tagging schemes to address overlapping entities, their decoding process is complex. Additionally, relying on ground truth hard tags for training can lead to overconfidence and hinder generalization [40].

The root cause of the aforementioned issues is that decomposition-based methods overlook a critical attribute of the triple: its subject, relation, and object are interdependent and inseparable. In other words, extracting one element without thoroughly understanding the information from the other two is not dependable. To address these issues, we attempt to approach the joint extraction task through the lens of triple classification. For instance, "Poutine" and "Canada" are two words in the sentence "Poutine is a dish made of fries and cheese curds from Canada." and "country" is a predefined relation, all of which can be observed in the training dataset. Intuitively, by judging its correctness, the triple (Poutine, Country, Canada) can be directly identified. This idea has three main advantages. First, subject, relation, and object are simultaneously fed into one classification module, fully capturing dependencies among the triple elements, thus reducing redundant information. Second, the direct design of a single-module, single-stage framework makes the network simple and facilitates easier training. Third, utilizing only one classification step effectively avoids cascading errors.

Motivated by the idea mentioned above, this study presents a novel joint extraction model, named FTC, adept at deriving all triples from unstructured text using a unified module and stage. To be specific, for a token pair (x_i, x_j) and a predetermined relation r_k, the classifier evaluates the validity of the triple(x_i, r_k, x_j). If valid, it assigns a meaningful tag; otherwise, it is tagged

"–". Consequently, for any input sentence, FTC outputs a three-dimensional matrix, where each element reflects the classification result for the combination (x_i, r_k, x_j). Experimental results on four commonly referenced public datasets demonstrate that the method we proposed outperforms earlier techniques and establishes new benchmark results.

The main contributions of this paper are as follows:

- We proposed a new end-to-end framework that makes it possible to simultaneously capture information about the subject, relation, and object, significantly mitigating the cascading error and relation redundancy issues.
- Aligned with our perspective, we introduce a new relation-specific upper triangular tagging strategy and a score-based classifier. The former ensures efficient decoding, while the latter assesses if a token pair combined with a relation constitutes a valid triple.
- We conduct extensive experiments on several public datasets. The results show that our approach surpasses state-of-the-art baselines, particularly in challenging situations involving overlapping triples.

2 Methodology

This section starts with a principled definition of relation triplet extraction, followed by a detailed description of each FTC model component. An overview of FTC is presented in Fig. 1.

2.1 Problem Definition

Given a sentence $S = \{x_1, x_2, ..., x_L\}$ with L tokens and K predefined relations $R = \{r_1, r_2, ..., r_K\}$. Finding all potential triplets $T(S) = \{(s, r, o) | s, o \in E, r \in R\}$ is the goal of joint entity and relation extraction, where E and R represent the sets of entities and relations. Subjects s and objects o are combinations of multiple contiguous tokens, that is, $entity.span = x_{p:q}$, where $x_{p:q}$ denotes the concatenation of tokens from x_p to x_q.

2.2 Relation-Specific Upper Triangular Tagging

For any provided sentence, we create a classifier that designates tags for all potential combinations of (x_i, r_k, x_j) where x_i, x_j belong to set S and r_k is part of set R. We keep a three-dimensional matrix $M^{L \times K \times L}$ to house these classification outcomes. Thus, we need to decode entities and relations from the matrix during evaluation.

Tagging. We utilize "BE" (Begin, End) symbols to express the details of a token's location within entities. Drawing from the understanding that entities can be identified by pinpointing their boundary tokens [26], our tagging strategy incorporates four types of tags: (1) SB-OB: This tag suggests that the two

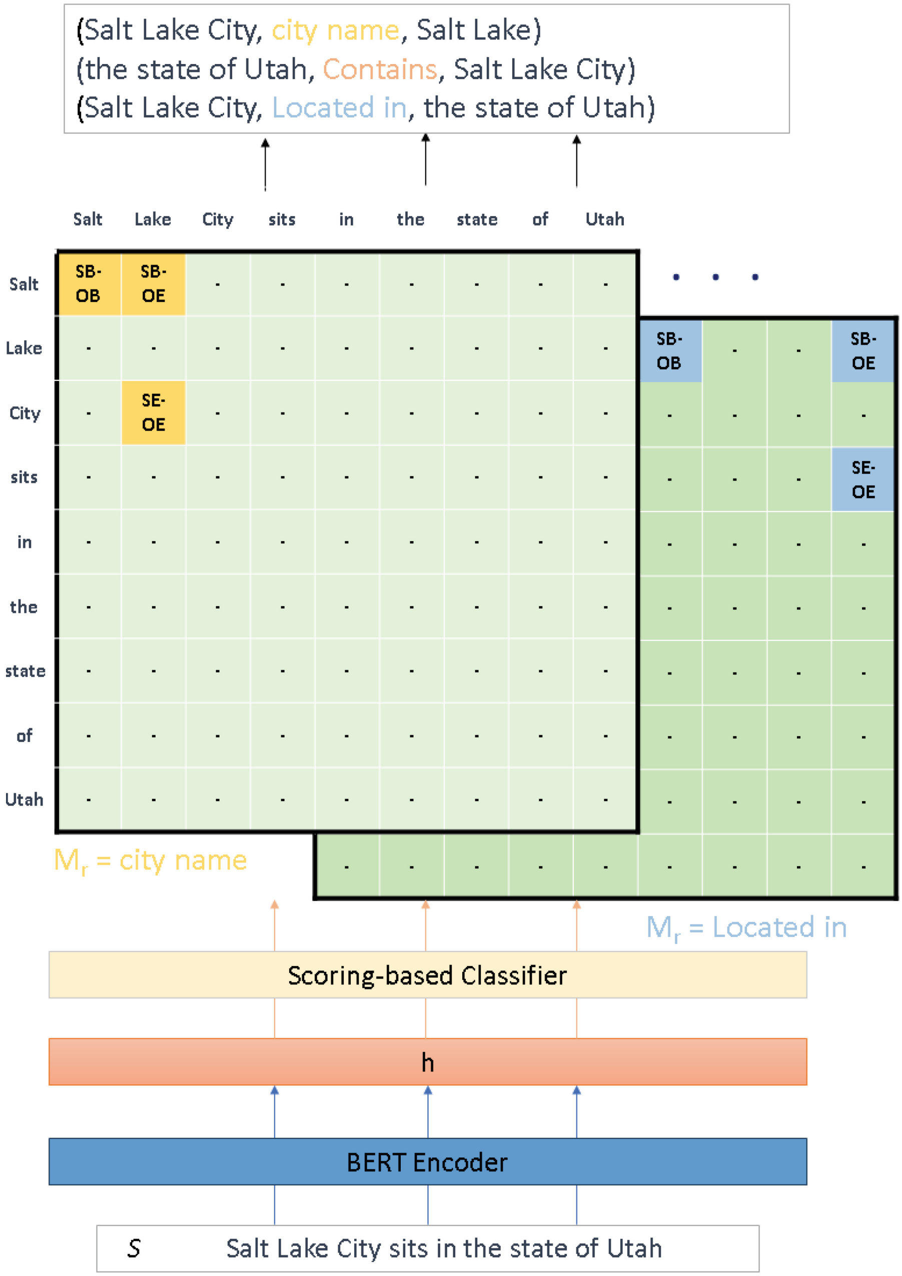

(Salt Lake City, city name, Salt Lake)
(the state of Utah, Contains, Salt Lake City)
(Salt Lake City, Located in, the state of Utah)
Salt
Lake
City
sits
in
the
state
of
Utah
SB-OB
SB-OE
SE-OE
SB-OB
SB-OE
SE-OE
. . .
M_r = city name
M_r = Located in
Scoring-based Classifier
h
BERT Encoder
S
Salt Lake City sits in the state of Utah

Fig. 1. The overall framework of the FTC. "SB" signifies the beginning token of the subject, while "OE" indicates the beginning token of the object.

positions correspond to the beginning tokens of the subject-object pair, given a definite relation. For example, given the relation “Located in” between “Salt Lake City” and “the state of Utah”, the combined classification tag (“Salt”, Located in, “the”) is assigned the tag “SB-OB”. (2) SB-OE: This tag indicates that the token linked to the row marks the subject’s start, whereas the token associated with the column signifies the object’s conclusion. For example, “Salt” is the beginning token of “Salt Lake City”, and “Utah” is the ending token of “the state of Utah”. Hence, the combination (“Salt”, Located in, “Utah”) is tagged as “SB-OE”. (3) SE-OE: This tag follows similar logic to “SB-OB”, implying that the two positions are the ending tokens of the paired subject and object. The combination (“City”, Located in, “Utah”) would be distribute “SE-OE”. (4) “–”: Apart from the above three cases, all cells are marked as “–”. As illustrated in Fig. 1, since the tagging only needs to be done on three corners of a rectangle, we aptly call this approach Relation-Specific Upper Triangular Tagging.

Obviously, the tagging matrix M is sparse, and this presents several advantages. First, only using three special tags reduces the potential search space during classification. Second, the sparsity of M provides ample negative samples during training and guarantees a streamlined and efficient decoding of triplets. Moreover, our relation-specific upper triangular tagging naturally addresses scenarios with overlapping patterns. In the EPO situation, the entity pairs are tagged in different submatrices based on their relations. In the case of SEO, when two triplets have an identical relation, the paired entities will be tagged differently within the same sub-matrix; otherwise, they are tagged in different submatrices. For the intricate SOO pattern, the entity pair, such as the triplet (Salt Lake City, City Name, Salt Lake), is located near the diagonal of its relation submatrix, allowing for easy decoding.

Decoding. The tagging matrix $M^{L\times K\times L}$ indicates the boundary tokens of the subjects and objects paired, along with their intervening relations. Thus, the decoding of triplets from M becomes straightforward and direct. Specifically, for each relation, the span of the subject is concatenated from “SB-OE” to “SE-OE”; the span of the object is concatenated from “SB-OB” to “SB-OE”; and the two paired entities share the same “SB-OE”.

2.3 Score-Based Classifier

To capture the token embedding $h = (e_1, e_2, ..., e_L)$ for each token in the input sentence, we use a pre-trained BERT model [4] as the sentence encoder.

$$h = BERT(x_1, x_2, ..., x_L) \tag{1}$$

where x_i represents each token’s input representation. It represents the total of the relevant token embedding and positional embedding.

Following this, we list all potential combinations of (e_i, r_k, e_j) and create a classifier to allocate tags with high confidence. Here, r_k represents the representation randomly initialized by the relation. In essence, this objective can be achieved using an ordinary classification network with the input (e_i, r_k, e_j). But, this network possesses two shortcomings: First, the simple classifier struggled to model the fundamental structural information of the triplets and failed to fully explore the connections between entities and relations. Second, classifying all combinations requires model to execute at least $L\times$ $\mathrm{K}\times L$ computations, which is time-consuming when utilizing (e_i, r_k, e_j) as the input.

Drawing from knowledge graph embedding techniques, we adopted ideas from RotatE [21]. Previous knowledge graph embedding models usually captured only a subset of the three relation patterns, i.e., symmetric/anti-symmetric, inversion, and composition, whereas RotatE can capture all relation patterns. The RotatE model characterizes each relation as a rotation from the source entity to the destination entity in the complex vector space. Here, we redefine the scoring function as the operation of non-linear concatenated projections:

$$x = d_r(h,t) = \phi(W[h;t]^T + b) \tag{2}$$

where W is the trainable weight matrix with dimensions $\mathbb{R}^{d_e\times 2d}$, while b stands for the trainable bias. Here, d_e represents the dimension of the entity pair representations. The function $\phi(.)$ denotes the ReLU activation and $[;]$ signifies the concatenation operation. This revised definition presents multiple benefits. First, our classifier scoring function can seamlessly connect with the sentence encoder's output. Second, the matrix W can be used to dynamically train the mapping function, which switches from entity attributes to the representation of entity pairs.

To ensure the model converges better during training, we normalize x to improve the model's generalization:

$$\varOmega = \frac{x-\mu[x]}{\sqrt{\sigma[x]+\epsilon}}\times\gamma+\beta \tag{3}$$

where μ and σ denote the mean and standard deviation of the data in that layer respectively; ϵ is a constant to ensure the denominator isn't zero; γ and β are learnable scaling and bias parameters respectively.

Then, we utilize all relation representations $R\in\mathbb{R}^{d_e\times 4K}$ to calculate the salience of token pairs (x_i, x_j) for all (x_i, r_k, x_j) where $k = 1, 2, ..., K$ and 4 represents the number of classification tags. Consequently, our method's final scoring function is defined as:

$$u(x_i, r_k, x_j)_{k=1}^{K} = W\times \mathrm{drop}(\varOmega) + b \tag{4}$$

where $W\in\mathbb{R}^{d_e\times 3d}$, b are trainable weights and biases, d_e indicates the dimension of entity pair representation, u is the scoring vector, and $drop(.)$ represents the dropout strategy [19], implemented to mitigate overfitting. Therefore, with only two fully connected layers, we achieve parallel scoring, reducing the actual processing steps to $L\times 1\times L$, which is even superior to TPLinker [23]. Moreover, the scoring function aligns with the RotatE philosophy, capturing correlations and exclusiveness between relations.

Finally, we apply the softmax function to the scoring vector of (x_i, r_k, x_j) to predict the associated tags:

$$P(y(x_i, r_k, x_j)|S) = \text{Softmax}(v(x_i, r_k, x_j)) \tag{5}$$

The objective function for FTC is defined as:

$$\begin{aligned} &\mathcal{L}_{triple} = -\tfrac{1}{L \times K \times L} \times \\ &\textstyle\sum_{i=1}^{L} \sum_{k=1}^{K} \sum_{j=1}^{L} \log P(y_{(x_i,r_k,x_j)} = g_{(x_i,r_k,x_j)}|\mathcal{S}) \end{aligned} \tag{6}$$

where $g(x_i, r_k, x_j)$ represents the gold tag obtained from annotations.

3 Experiments

3.1 Datasets and Evaluation Metrics

To ensure a fair and thorough comparison, we evaluated our model on the NYT [17] and WebNLG [6] datasets, following the procedures of [28] and [23]. These datasets have two different versions, which we represent as NYT*, NYT, WebNLG*, and WebNLG. It's important to highlight that NYT* and WebNLG* mark the final word of the entities, while NYT and WebNLG capture the full span of the entity span. To further investigate the proposed FTC's capability of handling complex scenarios, we additionally partitioned the test set based on overlap patterns and the number of triples per sentence. The statistics of the datasets are presented in Table 1.

Table 1. The dataset statistics for our experiments include information on N, which represents the number of triples in a sentence. A single sentence can simultaneously exhibit overlapping patterns of SEO, EPO, and SOO.

Dataset	Sentences				Details of test set						
	Train	Valid	Test	Relations	Normal	SEO	EPO	SOO	$N = 1$	$N > 1$	Triples
NYT*	56,195	4,999	5,000	24	3,266	1,297	978	45	3,244	1,756	8,110
WebNLG*	5,019	500	703	171	245	457	26	84	266	437	1,591
NYT	56,196	5,000	5,000	24	3,071	1,273	1,168	117	3,089	1,911	8,616
WebNLG	5,019	500	703	216	239	448	6	85	256	447	1,607

Following the previous works mentioned, during evaluation, we adopted partial matching for NYT* and WebNLG*: a triple is correct only if the relation and the last words of both subject and object are accurate. For NYT and WebNLG, we used exact matching: a predicted triple is deemed correct only if the entire spans of both entities and the relation match accurately. At the same time, we report standard precision (Prec.), recall (Rec.), and F1 scores for all baselines.

3.2 Implementation Details

Experiments were carried out on a workstation equipped with an Intel Xeon Gold 6240R 2.40 GHz CPU, 256 GB memory, an NVIDIA Tesla A100 40G GPU and CentOS 7.6 as the operating system. We implemented the model using PyTorch, optimizing all parameters with Adam [8] at a 1e-5 learning rate. For NYT and WebNLG, the batch sizes were set at 8 and 6, respectively. We used the pre-trained BERT base English model from Huggingface[1], consisting of 12 Transformer blocks with a 768-dimensional hidden size. The hidden layer dimension de was 3 times d. The maximum sequence length was 100, and dropout probability in Eq. (4) was set to 0.1.

3.3 Baselines

We evaluated our model against several relational triple extraction models:

(1) RSAN [30] uses relation-based attention mechanisms to create specific textual representations for every predefined relation, followed by entity recognition.
(2) CasRel [26] is a sequence labeling model employing cascade binary tagging framework to learn a relation-specific tagger for predefined relation.
(3) TPLinker [23] introduces a handshake marking scheme for simultaneous extraction of relations and entities during decoding.
(4) CasDE [12] employs a cascade dual decoder approach, beginning with relation identification within a sentence, followed by entity recognition.
(5) PRGC [38] completes extraction in three steps: predicting potential relations, extracting corresponding entities, and matching them to generate triples.
(6) OneRel [18] can directly extract triples from text sentences and then judge the correctness of the triples, effectively alleviating cascading errors and redundancy of entities.
(7) SPN [20] considers joint entities and relations extraction as a problem of predicting a direct set, avoiding the burden of predicting triple order.
(8) RS-TTS [33] uses a relation judgment module, a boundary smoothing mechanism, and an efficient tagging and scoring strategy to decode entities.
(9) ERFD-RTE [2] detects potential relations first, then carries out entity recognition for every relation to address overlapping triples and reduce redundant calculations.
(10) SOIRP [3] introduces a novel table filling method and decoding strategy to identify entities and align boundary tokens for every predefined relation, offering improved accuracy and efficiency.

One should note that RSAN utilize LSTM networks as their sentence encoders, whereas the other baseline models leverage a pre-trained BERT for feature representation. To ensure an equitable comparison, all results for these baselines are sourced directly from their original publications.

[1] https://huggingface.co/bert-base-cased.

3.4 Experimental Results and Analysis

Main Results. We compared the performance of our FTC model with 10 other baselines in the NYT and WebNLG datasets, as shown in Table 2, taking into account both partial and exact matching metrics. Evidently, FTC surpasses all 10 baselines, securing the top F1 performance for NYT, NYT*, WebNLG, and WebNLG*. We attribute FTC's stellar performance to two main advantages: First, FTC approaches the joint extraction challenge by viewing it as a detailed triplet classification endeavor. This enables simultaneous combination of entity and relation information during extraction, reducing redundant information. Second, combining the score-based classifier with relation-specific upper triangular tagging enables straightforward entity and relation extraction, effectively circumventing cascading errors.

Table 2. Precision, Recall and F1-score (%) of FTC and baselines model. **Bold** indicates the highest score.

Model	NYT*			WebNLG*			NYT			WebNLG		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
CasRel [26]	89.7	89.5	89.6	93.4	90.1	91.8	–	–	–	–	–	–
RSAN [30]	–	–	–	–	–	–	85.7	83.6	84.6	80.5	83.8	82.1
TPLinker [23]	91.3	92.5	91.9	91.8	92.0	91.9	91.4	92.6	92.0	88.9	84.5	86.7
CasDE [12]	90.2	90.9	90.5	90.3	91.5	90.9	89.9	91.4	90.6	88.0	88.9	88.4
PRGC [38]	93.3	91.9	92.6	94.0	92.1	93.0	93.5	91.9	92.7	89.9	87.2	88.5
OneRel [18]	92.8	92.9	92.8	94.1	94.4	94.3	93.2	92.6	92.9	91.8	90.3	91.0
SPN [20]	93.3	91.7	92.5	93.1	93.6	93.4	92.5	92.2	92.3	–	–	–
RS-TTS [33]	92.9	92.8	92.8	94.4	93.9	94.1	93.0	92.6	92.8	90.7	89.7	90.2
ERFD-RTE [2]	**94.0**	91.5	92.7	93.8	92.0	92.9	**94.0**	91.4	92.7	91.2	87.4	89.3
SOIRP [3]	93.3	92.9	93.1	94.3	94.1	94.2	93.4	92.6	93.0	**92.7**	89.6	91.2
FTC	93.3	**93.1**	**93.2**	**94.5**	**94.6**	**94.6**	93.6	**92.8**	**93.2**	92.4	**91.2**	**91.8**

Compared to the multi-module multi-stage approach PRGC, FTC secures absolute F1 score gains of 1.6 and 3.3 on WebNLG* and WebNLG, respectively. This implies that a unified step for extracting entities and relations can efficiently mitigate cascading error problems. Moreover, against other multi-module single-stage model, SOIRP, that uses four distinct tags to identify entities and relations, FTC surpasses it with absolute gains of 0.1, 0.4, 0.2 and 0.6 across the four datasets. These results confirm the efficacy of using one module to extract all triplets. These results verify that employing a single module to extract triplet elements in single step effectively captures the interplay between entities and relations elements, hinting that the single-module single-stage paradigm holds promise for the joint extraction task. Lastly, when compared with the single-module single-stage model OneRel, FTC outperforms OneRel across all evaluation metrics in every dataset. This suggests that the FTC is capable of recognizing more comprehensive triplets by capturing all three relationship patterns.

Detailed Results on Complex Scenarios. For validate the ability of FTC to handle overlapping patterns and multiple triplets, we carried out two yy tests on distinct subsets of WebNLG* and NYT*. We chose five robust methods as the baselines, with results showcased in Table 3.

Table 3. F1-score (%) for sentences with various overlapping patterns and numbers of triples. § denotes results from [38]. **Bold** marks the highest score.

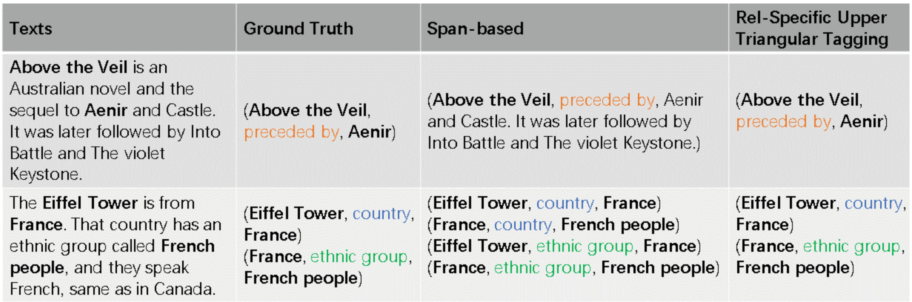
Model	NYT*									WebNLG*								
	Normal	SEO	EPO	SOO	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N \geq 5$	Normal	SEO	EPO	SOO	$N = 1$	$N = 2$	$N = 3$	$N = 4$	$N \geq 5$
TPLinker	90.1	94.0	93.4	90.1§	90.0	92.8	93.1	96.1	90.0	87.9	95.3	92.5	86.0§	88.0	90.1	94.6	93.3	91.6
CasRel	87.3	92.0	91.4	77.0§	88.2	90.3	91.9	94.2	83.7	89.4	94.7	92.2	90.4§	89.3	90.8	94.2	92.4	90.9
OneRel	90.6	95.1	94.8	90.8	90.5	93.4	93.9	96.5	94.2	91.9	95.4	94.7	94.9	91.4	93.0	95.9	95.7	94.5
PRGC	91.0	94.5	94.0	81.8	**91.1**	93.0	93.5	95.5	93.0	90.4	**95.9**	93.6	94.6	89.9	91.6	95.0	94.8	92.8
SOIRP	91.2	95.1	94.7	–	91.1	**93.7**	**94.4**	96.1	94.1	91.6	95.5	94.7	–	91.4	92.9	**96.0**	95.2	94.5
FTC	**91.2**	**95.3**	**95.0**	**90.9**	90.7	93.6	94.1	**96.5**	**94.4**	**92.2**	95.5	**94.9**	**95.1**	**91.5**	**93.2**	95.7	**95.7**	**94.8**

From the results, we can observe that FTC achieves the best F1 scores in 13 out of 18 subsets, especially in the most challenging scenarios: $N \geq 5$ and SOO. The SOO pattern encompasses two scenarios: one involves nested entities that most previous methods struggle to accurately recognize. Another scenario is where the subject and object share the same words. Moreover, sentences with $N \geq 5$ might simultaneously involve SEO, EPO, and SOO patterns, posing significant challenges to existing methods. However, our FTC manages to achieve optimal performance in SOO and $N \geq 5$ in both NYT* and WebNLG*, further attesting to the efficacy of our relation-specific upper triangular tagging. It inherently addresses the issue of overlapping triplets and proves more robust than the baselines in handling intricate scenarios.

Ablation Study. To assess the impact of each component on our model's performance, we conducted ablation experiments by removing specific components. The results are reported in Table 4.

When we replace the classifier with a simple linear network $f(x_i, r_k, x_j) = W[e_i; r_k; e_j] + b$, we found that the F1 score reduces by about 2.4% on NYT* and 3.5% on WebNLG*. The results prove that Score-based Classifier significantly enhances the model's overall performance. This mechanism enhances the classifier's classification ability by recognizing all three relation patterns and improves the extraction of relation elements in triples, highlighting the pivotal role of capturing dependencies between entities and relations during classifier design.

To compare with the sequence tagging scheme, we follow [26] and [23] by performing binary classification that identifies entity start and end positions using a span-based approach. In Table 4, when remove the Relation-Specific Upper Triangular Tagging strategy, the F1 is reduced by approximately 2.6% on NYT * and 4.1% on WebNLG *, highlighting the enhancement provided by our strategy. As shown in Fig. 2, the case study illustrates that span-based schemes often extract long entities and determine accurate subject-object pairs but overlook their interconnections due to focusing on entity positions rather than semantics. In contrast, the FTC tagging scheme excels in both areas, demonstrating superior generalizability and robustness.



Texts	Ground Truth	Span-based	Rel-Specific Upper Triangular Tagging
Above the Veil is an Australian novel and the sequel to **Aenir** and Castle. It was later followed by Into Battle and The violet Keystone.	(**Above the Veil**, preceded by, **Aenir**)	(**Above the Veil**, preceded by, Aenir and Castle. It was later followed by Into Battle and The violet Keystone.)	(**Above the Veil**, preceded by, **Aenir**)
The **Eiffel Tower** is from **France**. That country has an ethnic group called **French people**, and they speak French, same as in Canada.	(**Eiffel Tower**, country, **France**) (**France**, ethnic group, **French people**)	(**Eiffel Tower**, country, **France**) (**France**, country, **French people**) (**Eiffel Tower**, ethnic group, **France**) (**France**, ethnic group, **French people**)	(**Eiffel Tower**, country, **France**) (**France**, ethnic group, **French people**)

Fig. 2. Case study for the ablation of Relation-Specific Upper Triangular Tagging, using examples from the WebNLG* dataset. Correct entities are in **bold** and correct relations are highlighted in color.

Table 4. Ablation studies results (%) on NYT* and WebNLG* test datasets. w/o indicates a removed module. Bold marks the best result.

Model	NYT*			WebNLG*		
	Prec.	Rec.	F1	Prec.	Rec.	F1
ALL	**93.3**	**93.1**	**93.2**	**94.5**	**94.6**	**94.6**
w/o Classifier	91.3	90.5	90.8	90.8	91.4	91.1
w/o Tagger	90.5	90.7	90.6	89.7	91.3	90.5

Model Efficiency. We evaluated the model's efficiency by comparing the training and inference times of our model with the closely related baseline, TPLinker, on the WebNLG* and NYT* datasets. The results are presented in Table 5. In this experiment, both models were set with a batch size of 6 for training and 1 for testing, with the maximum input sentence length fixed at 100. Although both models have a theoretical complexity of $O(KL^2)$, FTC outperforms TPLinker in terms of parallel processing. Specifically, FTC processes K relations simultaneously, whereas TPLinker manages a single relation in each iteration. As a result, when the number of relations expands from 24 to 171, FTC's training time remarkably accelerates, being 1.3 times faster than TPLinker and soaring up to a whopping 4.1 times as fast. Unlike in training, FTC's F1 score significantly surpasses TPLinker's, demonstrating the efficiency of our introduced classifier. Moreover, FTC achieved a 4.6 times speed-up in inference time on WebNLG*, underscoring the effectiveness and rationale behind our proposed relation-specific upper triangular tagging.

Table 5. Efficiency comparison of the models. “Training Time (s)” represents the duration needed to train for one epoch, while “Inference Time (ms)” indicates the time taken to predict triples from a single sentence.

Dataset	Model	Training Time	Inference Time	F1-Score
NYT*	TPLinker	1592	46.2	91.9
	FTC	**1255**	**43.5**	**93.2**
WebNLG*	TPLinker	599	40.1	91.9
	FTC	**148**	**8.7**	**94.6**

4 Related Work

Based on the triple extraction procedure, existing joint procedures may be loosely classified into three categories:

Multi-module multi-stage modeling methods: These methods use varied modules and interconnected steps to sequentially extract entities and their corresponding relations. They can be categorized into three types. Entity Domain to Relation Domain Models: These first extract all entities from the text and then perform relation classification for each entity pair to finally obtain triples. Examples include models by [5,9,11]. Relation Domain to Entity Domain Models: Here, relations are first predicted from the text, and then entities (subject and object) are extracted on the basis of these relations. Models include those by [12,30,32,38]. Subject Domain to Relation/Object Domain Models: This approach extracts the subject and then infers the matched relation and object through sequence tagging or Q&A frameworks. Examples include models by [2,26,27] and [36]. Despite their success, these methods still suffer from cascading error issues as mistakes made in earlier steps can’t be corrected in subsequent steps.

Multi-module single-stage modeling methods: Such methods simultaneously extract entities and relations, subsequently merging them to form triples. For example, the works of [13,23] and [24] interpret entity recognition and relation classification as a table-filling challenge, with every cell representing the interaction between two distinct words. On the other hand, [20] and [35] recast joint extraction task as a set prediction challenge, bypassing the need to consider the sequence of predicting multiple triples. However, because the limited interplay between entities and relations in separate recognition, these approaches struggle to fully grasp the interdependencies among predicted entities and relations, causing overlap when formulating a triple.

Single-module single-stage modeling methods: These extract triples directly from sentences. The classic model, Novel-Tagging by [34], devises a sophisticated tagging strategy that creates connections between entities and relations, and it also recognizes triples in a sentence in a single step. However, this method cannot manage overlapping scenarios as it assumes that each entity pair retains only one relation at most. While the model proposed by [18] can handle overlapping entity scenarios, it can only recognize symmetric/anti-symmetric, inversion relation patterns and fails to recognize composite relation patterns, thus leading to incomplete extracted triples.

5 Conclusion

In our paper, we introduce an novel joint model featuring a score-based classifier and a relation-specific upper triangular tagging strategy. This architecture enables the retrieval of triplets in a single step through a unified module, effectively addressing challenges related to cascading errors and redundant information. Our experimental results on public datasets showcase the superior performance of our model relative to state-of-the-art methods in diverse scenarios. Further analysis, in particular, confirms the efficacy of our model in handling sentences with overlapping and multiple relations. We plan to focus on simplification and optimization of the model in future research to improve its feasibility and scalability in practical applications.

References

1. Bunescu, R.C., Mooney, R.J.: A shortest path dependency kernel for relation extraction. In: Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp. 724–731. Vancouver, BC, October 2005. http://www.cs.utexas.edu/users/ai-lab?bunescu:emnlp05
2. Chen, J., Hu, J., Li, T., Teng, F., Du, S.: An effective relation-first detection model for relational triple extraction. Expert Syst. Appl. **238**, 122007 (2024)
3. Dai, Q., Yang, W., Wang, L., Wei, F., Tuo, M.: SOIRP: subject-object interaction and reasoning path based joint relational triple extraction by table filling. Neurocomputing **580**, 127492 (2024)
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (June 2019). https://doi.org/10.18653/v1/N19-1423, https://aclanthology.org/N19-1423
5. Fu, T.J., Li, P.H., Ma, W.Y.: GraphRel: modeling text as relational graphs for joint entity and relation extraction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1409–1418. Association for Computational Linguistics, Florence, Italy (July 2019). https://doi.org/10.18653/v1/P19-1136, https://www.aclweb.org/anthology/P19-1136
6. Gardent, C., Shimorina, A., Narayan, S., Perez-Beltrachini, L.: Creating training corpora for NLG micro-planning. In: 55th annual meeting of the Association for Computational Linguistics (ACL) (2017)
7. Katiyar, A., Cardie, C.: Going out on a limb: joint extraction of entity mentions and relations without dependency trees. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 917–928 (2017)

8. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) 3rd International Conference on Learning Representations, San Diego, CA, USA, Conference Track Proceedings (2015)
9. Kong, W., Xia, Y.: CARE: co-attention network for joint entity and relation extraction. In: Calzolari, N., Kan, M.Y., Hoste, V., Lenci, A., Sakti, S., Xue, N. (eds.) Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), pp. 2864–2870. ELRA and ICCL, Torino, Italia (May 2024). https://aclanthology.org/2024.lrec-main.255
10. Li, Q., Ji, H.: Incremental joint extraction of entity mentions and relations. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 402–412 (2014)
11. Liu, J., Chen, S., Wang, B., Zhang, J., Li, N., Xu, T.: Attention as relation: learning supervised multi-head self-attention for relation extraction. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 3787–3793 (2021)
12. Ma, L., Ren, H., Zhang, X.: Effective cascade dual-decoder model for joint entity and relation extraction. arXiv preprint arXiv:2106.14163 (2021)
13. Miwa, M., Bansal, M.: End-to-end relation extraction using LSTMs on sequences and tree structures. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1105–1116 (2016)
14. Miwa, M., Sasaki, Y.: Modeling joint entity and relation extraction with table representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1858–1869 (2014)
15. Pawar, S., Palshikar, G.K., Bhattacharyya, P.: Relation extraction: a survey. CoRR **abs/1712.05191** (2017). http://arxiv.org/abs/1712.05191
16. Ratinov, L., Roth, D.: Design challenges and misconceptions in named entity recognition. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning, pp. 147–155. Association for Computational Linguistics, Boulder, Colorado (June 2009). https://www.aclweb.org/anthology/W09-1119
17. Riedel, S., Yao, L., McCallum, A.: Modeling relations and their mentions without labeled text. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS (LNAI), vol. 6323, pp. 148–163. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15939-8_10
18. Shang, Y.M., Huang, H., Mao, X.L.: Onerel: joint entity and relation extraction with one module in one step. In: AAAI Conference on Artificial Intelligence (2022). https://api.semanticscholar.org/CorpusID:247362852
19. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
20. Sui, D., Zeng, X., Chen, Y., Liu, K., Zhao, J.: Joint entity and relation extraction with set prediction networks. IEEE Trans. Neural Netw. Learn. Syst. (2023)
21. Sun, Z., Deng, Z.H., Nie, J.Y., Tang, J.: Rotate: knowledge graph embedding by relational rotation in complex space. arXiv preprint arXiv:1902.10197 (2019)
22. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In: Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003, pp. 142–147 (2003). https://www.aclweb.org/anthology/W03-0419
23. Wang, Y., Yu, B., Zhang, Y., Liu, T., Zhu, H., Sun, L.: TPLinker: single-stage joint extraction of entities and relations through token pair linking. In: Proceedings of

the 28th International Conference on Computational Linguistics, pp. 1572–1582 (2020)

24. Wang, Z., Nie, H., Zheng, W., Wang, Y., Li, X.: A novel tensor learning model for joint relational triplet extraction. IEEE Trans. Cybern. (2023)
25. Wang, Z., Wen, R., Chen, X., Huang, S.L., Zhang, N., Zheng, Y.: Finding influential instances for distantly supervised relation extraction. In: Proceedings of the 29th International Conference on Computational Linguistics, pp. 2639–2650 (2022)
26. Wei, Z., Su, J., Wang, Y., Tian, Y., Chang, Y.: A novel cascade binary tagging framework for relational triple extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1476–1488. Association for Computational Linguistics, Online (July 2020). https://doi.org/10.18653/v1/2020.acl-main.136, https://aclanthology.org/2020.acl-main.136
27. Ye, H., et al, H.: Contrastive triple extraction with generative transformer. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 14257–14265 (2021)
28. Yu, B., et al.: Joint extraction of entities and relations based on a novel decomposition strategy. In: ECAI 2020, pp. 2282–2289. IOS Press (2020)
29. Yu, X., Lam, W.: Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In: Coling 2010: Posters, pp. 1399–1407 (2010)
30. Yuan, Y., Zhou, X., Pan, S., Zhu, Q., Song, Z., Guo, L.: A relation-specific attention network for joint entity and relation extraction. In: International Joint Conference on Artificial Intelligence (2021). International Joint Conference on Artificial Intelligence
31. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. J. Mach. Learn. Res. **3**(Feb), 1083–1106 (2003)
32. Zeng, D., Zhao, C., Li, D., Dai, J.: Bdcore: bidirectional decoding with co-graph representation for joint entity and relation extraction. Knowl.-Based Syst. **294**, 111781 (2024)
33. Zhang, J., Jiang, X., Sun, Y., Luo, H.: RS-TTS: a novel joint entity and relation extraction model. In: 2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pp. 71–76. IEEE (2023)
34. Zhang, M., Zhang, Y., Fu, G.: End-to-end neural relation extraction with global optimization. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 1730–1740 (2017)
35. Zhang, Z., Zhang, H., Wan, Q., Liu, J.: Entity-relation triple extraction based on relation sequence information. Expert Syst. Appl. **238**, 121561 (2024)
36. Zhao, K., Xu, H., Cheng, Y., Li, X., Gao, K.: Representation iterative fusion based on heterogeneous graph neural network for joint entity and relation extraction. Knowl.-Based Syst. **219**, 106888 (2021). https://doi.org/10.1016/j.knosys.2021.106888
37. Zhao, T., Yan, Z., Cao, Y., Li, Z.: A unified multi-task learning framework for joint extraction of entities and relations. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 16, pp. 14524–14531 (2021). https://doi.org/10.1609/aaai.v35i16.17707
38. Zheng, H., et al.: PRGC: potential relation and global correspondence based joint relational triple extraction. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 6225–6235. Association for Computational Linguistics, Online, August 2021. https://doi.org/10.18653/v1/2021.acl-long.486, https://aclanthology.org/2021.acl-long.486

39. Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., Xu, B.: Joint extraction of entities and relations based on a novel tagging scheme. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1227–1236. Association for Computational Linguistics, Vancouver, Canada, July 2017. https://doi.org/10.18653/v1/P17-1113, https://aclanthology.org/P17-1113
40. Zhu, E., Li, J.: Boundary smoothing for named entity recognition. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 7096–7108 (2022)

H2O2Net: A Novel Entity-Relation Linking Network for Joint Relational Triple Extraction

Liang Zhang[1,2] and Nan Zheng[1,2](✉)

[1] Institute of Automation, Chinese Academy of Sciences, Beijing, China
{zhangliang2022,nan.zheng}@ia.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China
zhangliang221@mails.ucas.ac.cn

Abstract. Joint extraction of entities and relations from unstructured texts is a crucial task in large-scale knowledge graph construction. Recent methods achieve promising performance but still suffer from some inherent limitations, such as ignorance of the importance of relations in linking entities, overreliance on alignment between entity pairs, and decoding inefficiency. To deal with such problems, we propose a novel joint extraction framework, which is based on (entity, relation) pair linking, a new perspective to solve joint extraction. The framework is called H2O2Net since its decoding process is similar to the decomposition of H_2O_2. Specifically, two identical components are designed to predict (head entity, relation) and (tail entity, relation) pairs respectively, which is exploited by a linking strategy to generate triples. Such relation plays a natural role of connection, which alleviates the dependency of entity pairs alignment. Experimental results on benchmarks demonstrate that H2O2Net achieves state-of-the-art performance with higher efficiency and delivers consistent performance gain on complex scenarios of different overlapping patterns and multiple triples.

Keywords: Joint extraction · Relation linking · Triple overlap problem · Knowledge graph

1 Introduction

In the realm of knowledge graph construction, the extraction of relational triples, defined as (head entity, relation, tail entity) or $(h,\ r,\ t)$, from unstructured texts is pivotal. According to the role of relation, the main perspectives for handling relational triple extraction tasks can be divided into two classes: *relation as discrete label* and *relation as category*.

Traditional methodologies often adopt a sequential pipeline strategy [2,25,31]. It includes two steps: named entity recognition and relation classification.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78495-8_23.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 365–382, 2025.
https://doi.org/10.1007/978-3-031-78495-8_23

	Texts	Triples
Normal	[Tim Cook], the CEO of [Apple Inc], comes to [Beijing], [China].	(Apple Inc, company CEO, Tim Cook) (China, contains, Beijing)
SEO	[Tom] and [Jerry] live in the [United States].	(Tom, live in, the United States) (Jerry, live in, the United States)
EPO	[Kate Winslet] was born and raised in [Berkshire], England.	(Kate Winslet, birth place, Berkshire) (Kate Winslet, live in, Berkshire)
HTO	[Leonardo [DiCaprio]] is a famous American actor.	(Leonardo DiCaprio, family name, DiCaprio)

Fig. 1. Examples of the *Normal*, *SingleEntityOverlap* (SEO), *EntityPairOverlap* (EPO) and *HeadTailOverlap* (HTO[1]) patterns. The overlapping entities are marked in blue. (HTO is also called SOO when a triple is represented by (subject, relation, object)). (Color figure online)

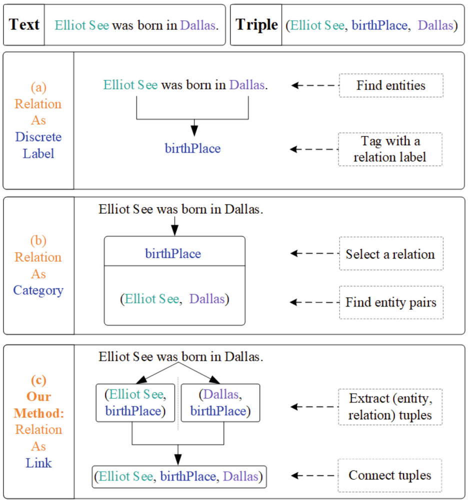

Text
Elliot See was born in Dallas.
Triple
(Elliot See, birthPlace, Dallas)
(a) Relation As Discrete Label
Elliot See was born in Dallas.
birthPlace
Find entities
Tag with a relation label
(b) Relation As Category
Elliot See was born in Dallas.
birthPlace
(Elliot See, Dallas)
Select a relation
Find entity pairs
(c) Our Method: Relation As Link
Elliot See was born in Dallas.
(Elliot See, birthPlace)
(Dallas, birthPlace)
(Elliot See, birthPlace, Dallas)
Extract (entity, relation) tuples
Connect tuples

Fig. 2. The main perspectives of existing methods and our method according to the role of relation. The solid arrow indicates the process of triple extraction and the dotted arrow explains the purpose of the target step.

The entities are firstly recognized and relation types are assigned for all candidate entity pairs. Such approaches treat relations as discrete labels that ignore the interdependence between entities and relations and suffer from error propagation. Different from pipeline methods, subsequent joint learning methods [5,27,29] aim to extract relational triples in an end-to-end way, which also regard relations as discrete labels and fail to solve the overlapping triple problem where multiple relational triples in a sentence share identical entities, as shown in Fig. 1. Both methods seldom consider the effect of relations or interaction between entities and relations (See Fig. 2(a)).

Recent models try to emphasize the semantics of relations [11,14] or treat relations as categories [18,20,21,28]. Some works tend to detect the relations in a sentence first, then extract entities for a specific relation so that the relation is similar to a category that contains associated entities (See Fig. 2(b)). These models neglect the interaction between relations and entities, as well as the link role of relations. Moreover, most of these models excessively rely on the alignment of entity pairs, so their performance will decline seriously when the alignment component is removed or the task of entity pair alignment is hard to learn.

To address the aforementioned problems, we aim to reduce the dependence on entity alignment through relations, which reduces information redundancy and improves decoding efficiency. Therefore, we model the relational triple extraction task as an (entity, relation) pair linking problem from a fresh perspective instead of treating relations as discrete labels or categories for entity pairs. For all possible head entities and tail entities in a sentence, all relations associated with each entity are identified before (head entity, relation) and (tail entity, relation) pairs are attained respectively. Finally, the two kinds of pairs with identical relations are connected to generate relational triples (See Fig. 2(c)). The whole process is just like the decomposition of H_2O_2 as follows:

$$2H_2O_2 = 2H_2O + O_2 \uparrow \tag{1}$$

where H represents the head entity or tail entity, and O represents the relation. For this reason, our model is called H2O2Net. Specifically, H2O2Net consists of a BERT-based encoder module, two entity-relation classification modules, and a link module. The main contributions of this paper are as follows:

1. We introduce a novel paradigm for relational triple extraction by taking the task as a problem of (entity, relation) pair linking. This approach enables our model to capture the nuanced interplay between entities and relations more effectively than conventional methods, which is facilitated by an innovative linking mechanism unique to our framework.
2. We present a joint extraction model that leverages (entity, relation) pair linking. This design significantly reduces model complexity, operating with fewer parameters on average compared to existing models while maintaining comparable accuracy.
3. Our experiments reveal that H2O2Net achieves top-tier performance with fewer parameters and less computational cost, significantly cutting down on FLOPs and speeding up inference. Across various benchmarks, our model sets new standards for efficiency and accuracy.

2 Related Work

Relational triple extraction tasks can be classified based on several criteria, such as extracting manner (pipelined or joint), extracting approach (generative or labeling), extracting procedure (multi-stage or one-stage), and so on. Furthermore, these models can be categorized into two classes based on their conceptualization of relations:

The first class is *relation as discrete label*, which treats relations as discrete labels of entity pairs or entities. For example, For instance, one approach employs relation classification (RC) on pairs of extracted entities after named entity recognition (NER) is done [2,25,30,31]. The pipelined methods usually predict which relation label the entity pairs may own in the RC stage. The second line of works extract entities and relations in a joint model via sequence labeling or triples generating. The former approach acknowledges the role of relations between entities within the annotations [1,3,23,29], and the latter predicts the relations of the entity pairs in the entity generation phase [4,5,15,19,26,27]. Despite their successes, methods treating the relation as a discrete label pay limited attention to the dynamics of relations or the interaction between entities and relations, which leads to redundancy during sequence decoding or triple construction. Meanwhile, most of them are susceptible to the problem of error propagation and low efficiency.

The second class is *relation as category*, which treats relations as categories containing related entities. For example, the first line of works extract entities or entity pairs for each relation which consider all relations in the relation set [1,12,16,18,20,21,24]. A relation is considered existent if the corresponding entities or entity pairs are identified, and non-existent if not. The second approach initially identifies all possible relations conveyed in a sentence rather than preserve all redundant relations, then extract entity pairs according to potential relations [11,14,28]. Such *relation as category* methods alleviate the problems of redundancy relations but still ignore the interaction between relations and entities as well as the link role of relations. Additionally, certain methods rely heavily on the alignment of entity pairs, which makes them unsuitable when handling large texts due to the huge computational complexity of the entity pairs alignment module.

Different from existing methods, in this paper, we aim to treat the relational triple extraction task as an (entity, relation) pair linking problem where the relation plays a natural role of connection. As a result, this approach efficiently capitalizes on the interdependence between entities and relations, enhancing efficiency and reducing information redundancy. Because the (entity, relation) pair linking is executed in a single stage, the aforementioned error propagation and overdependence of entity pairs alignment can be greatly addressed.

3 Method

In this section, we first start with a principled problem definition, and then elaborate on each component of H2O2Net. An overview illustration of H2O2Net is shown in Fig. 3.

3.1 Problem Definition

Given a sentence $\mathcal{S} = \{w_1, w_2, \ldots, w_L\}$ with L tokens and K predefined relations $\mathcal{R} = \{r_1, r_2, \ldots, r_K\}$. The desired output of joint relational triple extraction is

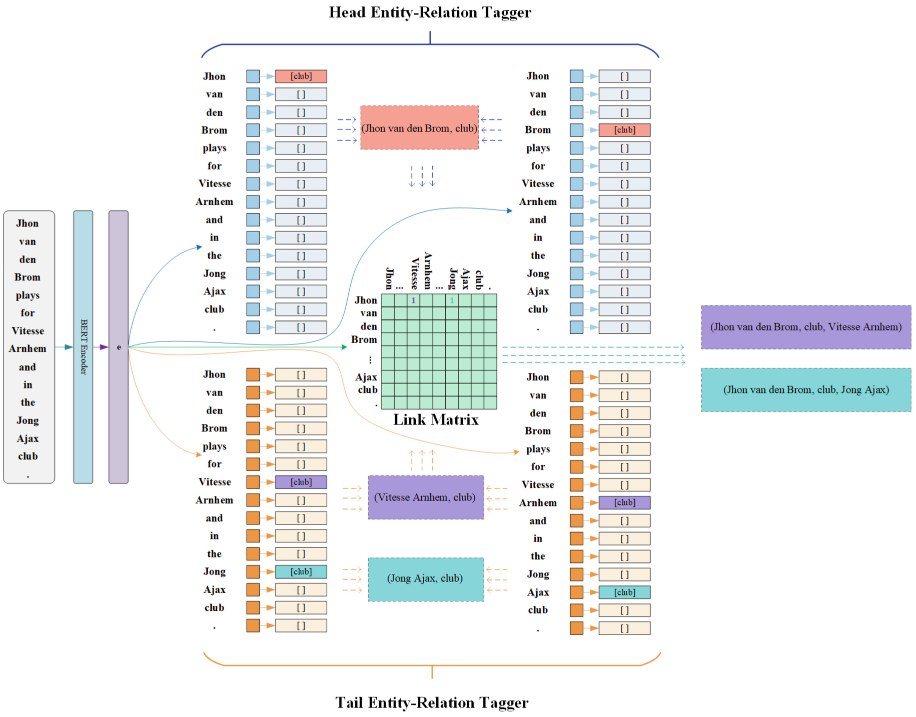

Head Entity-Relation Tagger
Jhon
van
den
Brom
plays
for
Vitesse
Arnhem
and
in
the
Jong
Ajax
club
.
BERT Encoder
e
[club]
[]
(Jhon van den Brom, club)
Link Matrix
1
(Vitesse Arnhem, club)
(Jong Ajax, club)
(Jhon van den Brom, club, Vitesse Arnhem)
(Jhon van den Brom, club, Jong Ajax)
Tail Entity-Relation Tagger

Fig. 3. An overview of the proposed H2O2Net. Given a sentence, H2O2Net predicts all (head entity, relation) pairs, i.e. (*Jhon van den Brom, club*), by the Head Entity-Relation Tagger and all (tail entity, relation) pairs, i.e. (*Vitesse Arnhem, club*) and (*Jong Ajax, club*), by the Tail Entity-Relation Tagger. Then, for each (head entity, relation) pair and (tail entity, relation) pair with an identical relation, if the token pair, consisting of the first tokens of the two entities, e.g. (*Jhon, Vitesse*) or (*Jhon, Jong*), is tagged with 1 in the Link Matrix, the triple will be extracted by connecting the matched pairs.

to identify all possible triples as $\mathcal{T} = \{(h, r, t)\,|r \in \mathcal{R}\}$, where $h,\ t$ are the head entity and tail entity composed of several consecutive tokens, i.e. $ent = w_i : w_j$, where ent is h or t, $w_i : w_j$ refers to the concatenation of w_i to w_j.

3.2 H2O2Net Encoder

The output of H2O2Net Encoder is $\mathcal{E} = \{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_L | \boldsymbol{e}_i \in \mathbb{R}^{d \times 1}\}$, where d is the embedding dimension, and L is the number of tokens. A pre-trained BERT model [8] is applied to encode the input sentence for fair comparison, but theoretically it can be extended to other encoders, e.g. ALBert [10], for low-cost devices.

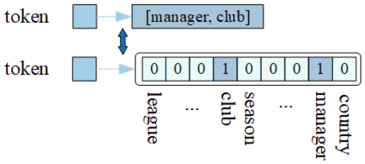


Fig. 4. One method to solve the multi-label multi-classification problem. In this example, to predict the *manager* and *club* relation for a *token*, this method aims to estimate the probability of the existence for the relation *club* and *manager*.

3.3 H2O2Net Decoder

In this section, we describe our instantiation of the entity-relation tagging scheme.

Entity-Relation Tagger. The Entity-Relation Taggers, which are shown as the blue and orange boxes in Fig. 3, are designed to recognize all possible (entity, relation) pairs in the input sentence. Both Head Entity-Relation Tagger (denoted as Head-ERT) and Tail Entity-Relation Tagger (denoted as Tail-ERT) function identically, differing only in their learned parameters, thus we will focus on the mechanics of Head-ERT for clarity.

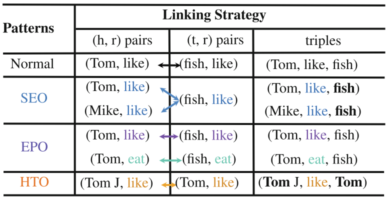
Head-ERT treats the (head entity, relation) pair extraction as a multi-label multi-classification task. It adopts two identical multi-label multi-class classifiers to distinguish all possible relations for each token. Although one classifier is enough to detect the span of a head entity by tagging consecutive tokens with an identical relation label, it suffers from the problem of overlapping triples. Therefore, we adopt two classifiers instead, which detect the start token and the end token for a head entity in parallel by an identical tagged relation label.

As illustrated in Fig. 4, to solve the multi-label multi-classification problem, for each token, a set of binary classifiers is adopted to estimate the probability of existence for each relation, respectively. The corresponding relation will be assigned with tag 1 if the probability exceeds a certain threshold λ_1 or tag 0 otherwise. Each binary classifier is initiated with a one-layer fully connected network. The detailed operations of Head-ERT on each token and relation are as follows:

$$p_{ij}^{hs} = \sigma\left(\boldsymbol{W}_j^{hs}\boldsymbol{e}_i + \boldsymbol{b}_j^{hs}\right) \tag{2}$$

$$p_{ij}^{he} = \sigma\left(\boldsymbol{W}_j^{he}\boldsymbol{e}_i + \boldsymbol{b}_j^{he}\right) \tag{3}$$

where p_{ij}^{hs} and p_{ij}^{he} represent the probability of identifying the i-th token in the input sentence as the start and end position of a head entity with the j-th relation existing in the token, respectively. $\boldsymbol{W}_{(\cdot)} \in \mathbb{R}^{d\times 1}$ is the trainable weight, $\boldsymbol{b}_{(\cdot)}$ is the bias and σ is the sigmoid activation function.



Patterns	Linking Strategy		
	(h, r) pairs	(t, r) pairs	triples
Normal	(Tom, like)	(fish, like)	(Tom, like, fish)
SEO	(Tom, like) (Mike, like)	(fish, like)	(Tom, like, **fish**) (Mike, like, **fish**)
EPO	(Tom, like) (Tom, eat)	(fish, like) (fish, eat)	(Tom, like, fish) (Tom, eat, fish)
HTO	(Tom J, like)	(Tom, like)	(**Tom** J, like, **Tom**)

Fig. 5. The linking strategy for different overlapping patterns. The *h*-*r* pair and *t*-*r* pair represent the (head entity, relation) pair and (tail entity, relation) pair respectively.

Head-ERT optimizes the following likelihood function to identify the span of head entity h and relation r given a sentence representation s:

$$p_{\theta_h}\left((h,r)\,|s\right) = \prod_{m\in\{hs,he\}} \prod_{i=1}^{L}\prod_{j=1}^{K}\left(p_{ij}^{m}\right)^{\mathbf{I}\{y_{ij}^{m}=1\}}\left(1-p_{ij}^{m}\right)^{\mathbf{I}\{y_{ij}^{m}=0\}} \tag{4}$$

where L is the length of the tokens and K is the number of the relations. $\mathbf{I}\{z\} = 1$ if z is true and 0 otherwise. y_{ij}^{hs}/y_{ij}^{he} is the binary tag of head entity start/end position for the i-th token in s with j-th relation in $\mathcal{R}$. The parameters $\theta_h = \{\boldsymbol{W}^{hs}, \boldsymbol{b}^{hs}, \boldsymbol{W}^{he}, \boldsymbol{b}^{he}\}$. Likewise, for the Tail-ERT, we obtain p_{ij}^{ts}, p_{ij}^{te} and $p_{\theta_t}((t,r)|s)$ as follows:

$$p_{ij}^{ts} = \sigma\left(\boldsymbol{W}_j^{ts}\boldsymbol{e}_i + \boldsymbol{b}_j^{ts}\right) \tag{5}$$

$$p_{ij}^{te} = \sigma\left(\boldsymbol{W}_j^{te}\boldsymbol{e}_i + \boldsymbol{b}_j^{te}\right) \tag{6}$$

$$p_{\theta_t}\left((t,r)\,|s\right) = \prod_{m\in\{ts,te\}} \prod_{i=1}^{L}\prod_{j=1}^{K}\left(p_{ij}^{m}\right)^{\mathbf{I}\{y_{ij}^{m}=1\}}\left(1-p_{ij}^{m}\right)^{\mathbf{I}\{y_{ij}^{m}=0\}} \tag{7}$$

For multiple entities detection, we adopt the nearest start-end pair match principle to decide the span of any entities. For example, as shown in Fig. 3, for the relation *club*, the nearest end token to the start token *Vitesse* is *Arnhem*, hence the detected result of the entity will be *Vitesse Arnhem*. Meanwhile, the extracted (entity, relation) pair is (*Vitesse Arnhem, club*) Note that the end token can not appear before the given token.

The two Entity-Relation Taggers work in parallel to extract (entity, relation) pairs expressed by each sentence. The complex scenarios with overlapping patterns are addressed naturally as shown in Fig. 5. For the EPO case, the pairs with the same relation will be connected. For instance, (*Tom, like, fish*) and (*Tom, eat, fish*) are two EPO triples, thus, they are connected by the *like* in purple and *eat* in cyan in Fig. 5. For the SEO case, the linking strategy is similar to EPO when the relations of triples are different. Otherwise, the (tail entity, relation) pair or (head entity, relation) pair, e.g. the (*fish, like*) pair with the *like* in blue, will be reused as shown in Fig. 5. For the HTO scenario, e.g. the triple (*Tom J, like, Tom*) in Fig. 5, there is no overlap because the head entity and tail entity are extracted by different taggers.

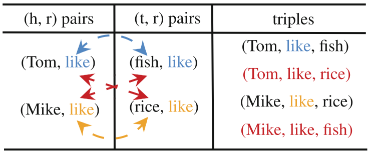


Fig. 6. The triples in red are the unreliable triples that are mismatched by the (head entity, relation) pair and the (tail entity, relation) pair. (Color figure online)

Table 1. Statistics of datasets. N is the number of triples in a sentence.

Category	Dataset				Details of Test Set									
	Train	Valid	Test	Relations	Normal	SEO	EPO	HTO	N = 1	N = 2	N = 3	N = 4	N > 5	Triples
NYT*	56195	4999	5000	24	3266	1297	978	45	3244	1045	312	291	108	8110
WebNLG*	5019	500	703	171	245	457	26	84	266	171	131	90	45	1591
NYT	56195	5000	5000	24	3222	1273	969	117	3240	1047	314	290	109	8120
WebNLG	5019	500	703	216	239	448	6	85	256	175	138	93	41	1607

However, when there is more than one (h, r) pair shared an identical relation with more than one (t, r) pair, this method may be confused. For example, as shown in Fig. 6, two triples (*Tom, like, fish*) and (*Mike, like, rice*) tend to be reconstructed by two (head entity, relation) pairs, i.e. (*Tom, like*) and (*Mike, like*), and two (tail entity, relation) pairs, i.e. (*fish, like*) and (*rice, like*), but these pairs may also generate other unreliable triples, i.e. (*Tom, like, rice*) and (*Mike, like, fish*). To handle such a situation, we introduce an alignment module of entity pairs, named Link Matrix Module. Note that we can exploit some heuristic methods to filter out the unreliable triples or just ignore the unreliable generation problem for the rarity of this situation in an application.

Link Matrix Module. To establish accurate connections between (entity, relation) pairs, we introduce the Link Matrix, as shown in the green matrix in Fig. 3. Note that the Link Matrix can be learned simultaneously with the Entity-Relation Taggers since these two stages are independent of each other. The detailed process is as follows: first, all possible token pairs in a sentence are enumerated; then the pair whose link score exceeds a certain threshold λ_2 is selected. The Link Matrix is similar to a digraph, and each token of an entity can be one vertex of the link edge. Here we choose the first token of the entity, which means a head entity and a tail entity can be connected if the token pair constructed by their first tokens is selected.

Given a sentence with L tokens, the size of the link matrix will be $\mathbb{R}^{L \times L}$. The link score of each element of the matrix is obtained as follows:

$$p_{ij}^{l} = \sigma\left(\boldsymbol{W}^{l2}\ \phi\left(\boldsymbol{W}^{l1}[\boldsymbol{e}_i; \boldsymbol{e}_j] + \boldsymbol{b}^{l1}\right) + \boldsymbol{b}^{l2}\right) \tag{8}$$

where $[;]$ is the concatenation operation. $\boldsymbol{W}^{l1} \in \mathbb{R}^{2d \times d}$ and $\boldsymbol{W}^{l2} \in \mathbb{R}^{d \times 1}$ are the trainable weights, $\boldsymbol{b}_{(\cdot)}$ is the bias, σ is the sigmoid activation function and

ϕ is the ReLU activation function. The link score will be high if the token pair, which is constructed by the first tokens, is in a triple.

3.4 Training Strategy

The model is trained jointly and the parameters of the H2O2Net encoder are shared. The loss of the Entity-Relation Taggers is:

$$\mathcal{L}_{main} = -(\log p_{\theta_h}((h, r)|s) + \log p_{\theta_t}((t, r)|s)) \tag{9}$$

The loss of the link matrix module is:

$$\mathcal{L}_{link} = -\sum_{i=1}^{L}\sum_{j=1}^{L}(y_{ij}\log p_{ij}^{l} + (1 - y_{ij})\log(1 - p_{ij}^{l})) \tag{10}$$

where y_{ij} is the gold tag for the i-th token and the j-th token which represent the first token of the head entity and tail entity respectively. The total loss is the sum of the above two losses:

$$\mathcal{L}_{total} = \mathcal{L}_{main} + \alpha\mathcal{L}_{link} \tag{11}$$

where α is the weight of the Link Matrix Module. The model is trained by minimizing $\mathcal{L}_{total}$ through Adam [9] optimizer over shuffled mini-batches.

4 Experiments

4.1 Datasets

Consistent with prior research [18,20,21,28], our model is evaluated on two public datasets NYT [17] and WebNLG [7], both of which have two versions: one version only annotates the last word of entities, and the other version annotates the whole span of entities. In this paper, the first version datasets are denoted as NYT* and WebNLG*, and the second version datasets are denoted as NYT and WebNLG. To further study the capability of H2O2Net in extracting overlapping and multiple relations, the test set is split by overlapping patterns and triple numbers. The detailed statistics are reported in Table 1.

4.2 Evaluation Metrics

Three standard evaluation metrics are used in our experiments, i.e. micro Precision (Prec.), Recall (Rec.), and F1-score (F1). During the evaluation, we adopt *Partial Match* for NYT* and WebNLG*, and *Exact Match* for NYT and WebNLG. An extracted triple is regarded as correct only if it is a correct match with ground truth, which means the last word of entities (for NYT* and WebNLG*) or the whole entity spans (for NYT and WebNLG) of both head and tail entities and the relations are all correct.

Table 2. Precision(%), Recall(%) and F1-score(%) of our proposed H2O2Net and baselines. **Bold** marks the highest score and ‡ means the results are produced by us with the available source code.

Model	NYT*			WebNLG*			NYT			WebNLG		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
ETL-span [23]	84.9	72.3	78.1	84.0	91.5	87.6	85.5	71.7	78.0	84.3	82.0	83.1
RSAN [24]	–	–	–	–	–	–	85.7	83.6	84.6	80.5	83.8	82.1
CasRel [21]	89.7	89.5	89.6	93.4	90.1	91.8	–	–	–	–	–	–
TPLinker [20]	91.3	**92.5**	91.9	91.8	92.0	91.9	91.4	92.6	92.0	88.9	84.5	86.7
CasDE [14]	90.2	90.9	90.5	90.3	91.5	90.9	89.9	91.4	90.6	88.0	88.9	88.4
PRGC [28]	**93.3**	91.9	92.6	94.0	92.1	93.0	**93.5**	91.9	92.7	89.9	87.2	88.5
EmRel [22]	91.7	**92.5**	92.1	92.7	93.0	92.9	92.6	**92.7**	92.6	90.2	87.4	88.7
OneRel‡ [18]	91.9	92.3	92.1	93.2	92.9	93.0	91.4	92.2	91.8	90.0	88.0	89.0
SPN [19]	**93.3**	91.7	92.5	93.1	93.6	93.4	92.5	92.2	92.3	–	–	–
ERGM [6]	**93.3**	91.5	92.4	94.2	91.2	92.7	–	–	–	–	–	–
H2O2Net	**93.3**	92.3	**92.8**	**94.5**	**94.1**	**94.3**	**93.5**	92.3	**92.9**	**92.7**	**89.7**	**91.2**

4.3 Implementation Details

In our experiments, the entire training process was conducted on a workstation equipped with an Intel Xeon E5 2.20 GHz CPU, 503 GB of memory, a single NVIDIA Tesla V100-SXM2 GPU, and Ubuntu 18.04. To ensure a fair comparison, we utilized the BERT-Base-Cased English model provided by Huggingface[1], setting the maximum input sentence length to 100, in line with previous studies [21,28].

The model is implemented with PyTorch and the parameters are optimized by Adam [9] with a batch size of 64/6 for NYT/WebNLG. The encoder learning rate for BERT is set as 1×10^{-4}, and the downstream learning rate is set as 1×10^{-3}. For simplicity, the weight of the Link Matrix Module (α) is set to 1 and the threshold of the Entity-Relation Taggers (λ_1) and the Link Matrix Module (λ_2) are both set to 0.5, but the performance might be better by carefully tuning α, λ_1 and λ_2. The model is trained for 100 epochs and the best model is chosen to output results on the test set. We also conduct weight decay [13] with a rate of 0.01 and adopt an early stopping mechanism to prevent the model from overfitting. Other hyper-parameters are determined on the validation set.

4.4 Baseline Models

For comparison, the following ten models are employed as baselines: ETL-span [23], RSAN [24], CasRel [21], CasDE [14], TPLinker [20], PRGC [28], EmRel [22], OneRel‡ [18], SPN [19] and ERGM [6]. Note that the sentence encoders used in ETL-span and RSAN are LSTM networks, while other baselines employ a pre-trained BERT to obtain feature representations. All the experimental results of the baseline models are directly taken from the original literature unless specified.

[1] https://huggingface.co/bert-base-cased.

Table 3. F1-score(%) on sentences with different overlapping patterns and triple numbers. **Bold** marks the highest score, § marks the results reported by [28] and ‡ means the results are produced by us with the available source code.

Model	NYT*									WebNLG*								
	Normal	SEO	EPO	HTO	N = 1	N = 2	N = 3	N = 4	N>5	Normal	SEO	EPO	HTO	N = 1	N = 2	N = 3	N = 4	N>5
CasRel	87.3	91.4	92.0	77.0§	88.2	90.3	91.9	94.2	83.7	89.4	92.2	94.7	90.4§	89.3	90.8	94.2	92.4	90.9
TPLinker	90.1	93.4	94.0	90.1§	90.0	92.8	93.1	96.1	90.0	87.9	92.5	95.3	86.0§	88.0	90.1	94.6	93.3	91.6
PRGC	**91.0**	94.0	94.5	81.8	**91.1**	93.0	93.5	95.5	**93.0**	90.4	93.6	**95.9**	94.6	89.9	91.6	95.0	94.8	92.8
OneRel‡	89.9	93.9	94.6	83.9	89.8	93.1	93.8	96.0	91.8	90.5	93.6	95.1	93.6	90.0	92.2	94.8	93.7	94.1
H2O2Net	90.9	**94.4**	**94.8**	**90.3**	90.9	**93.4**	**93.9**	**96.3**	92.4	**92.0**	**94.7**	**95.9**	**94.8**	**91.5**	**93.2**	**96.3**	**95.2**	**94.6**

Table 4. Comparison of model efficiencies on NYT* and WebNLG* datasets. Results of other methods are obtained by the official implementation with the default configuration. Params$_{decoder}$ are calculated on the decoder. Inference Time (ms) is the mean time to predict triples for one sentence.

Dataset	Model	Complexity	FLOPs (G)	Params$_{decoder}$	Inference Time (1/32)	F1-score
NYT*	PRGC	$\mathcal{O}(L^2)$	24.082	3851167	15.6/5.3	92.6
	OneRel‡	$\mathcal{O}(KL^2)$	300.810	3762528	36.0/62.2	92.1
	H2O2Net	$\mathcal{O}(L^2 + KL)$	**23.623**	**1255009**	**14.9**/**5.2**	**92.8**
WebNLG*	PRGC	$\mathcal{O}(L^2)$	24.082	3907762	15.7/**6.0**	93.0
	OneRel‡	$\mathcal{O}(KL^2)$	409.190	5117868	35.4/46.2	93.0
	H2O2Net	$\mathcal{O}(L^2 + KL)$	**23.713**	**1707181**	**14.4**/**6.0**	**94.3**

4.5 Experimental Results

Overall Results. Table 2 shows the results of our model against baseline models on NYT and WebNLG in terms of *Partial Match* and *Exact Match*. Our H2O2Net model outperforms them with respect to almost all evaluation metrics.

Compared with the representative *relation as discrete label* method SPN, H2O2Net obtains 0.3, 0.9, and 0.6 absolute gains in F1-score on three datasets, respectively. This demonstrates the importance of interaction between entities and relations. Compared with the *relation as category* model PRGC, H2O2Net obtains 0.2, 1.3, 0.2, and 2.7 absolute gains in F1-score on four datasets, respectively. Even if compared with another recent strong *relation as category* model OneRel‡, H2O2Net obtains 0.7, 1.3, 0.3, and 2.2 absolute gains in F1-score on four datasets, respectively. Such results indicate that considering the link role of relations is important in exploring the interactions between entities and relations. These all indicate that the *relation as link* is effective for the joint relational triple extraction task.

Overall, H2O2Net shows competitive performance with the existing state-of-the-art models. It outperforms OneRel‡ by 2.2% on WebNLG, which is regarded as the most difficult dataset to handle [20]. Moreover, H2O2Net achieves higher Precision scores than Recall scores on four datasets, which means H2O2Net is capable of extracting accurately and suitable for the scenarios requiring accuracy. We attribute the high Precision of H2O2Net to its effective Linking Strategy and Link Matrix Module, which connect the reliable (entity, relation) pairs and filter out the invalid triples, respectively.

Table 5. Results (%) on different subtasks. (h, t) denotes the entity pair and r means the relation. ‡ means the results are produced by us with the available source code.

Model	Element	NYT*			WebNLG*		
		Prec.	Rec.	F1	Prec.	Rec.	F1
CasRel	(h, t)	89.2	90.1	89.7	95.3	91.7	93.5
	r	96.0	93.8	94.9	96.6	91.5	94.0
	(h, r, t)	89.7	89.5	89.6	93.4	90.1	91.8
PRGC	(h, t)	**94.0**	92.3	**93.1**	96.0	93.4	94.7
	r	95.3	**96.3**	95.8	92.8	**96.2**	94.5
	(h, r, t)	**93.3**	**91.9**	92.6	94.0	92.1	93.0
OneRel‡	(h, t)	92.0	**92.9**	92.4	95.1	94.6	94.9
	r	96.2	95.2	95.7	96.2	94.6	95.4
	(h, r, t)	91.9	**92.3**	92.1	93.2	92.9	93.0
H2O2Net	(h, t)	93.2	92.8	93.0	**96.3**	**95.8**	**96.0**
	r	**97.0**	95.3	**96.1**	**96.9**	95.7	**96.3**
	(h, r, t)	**93.3**	**92.3**	**92.8**	**94.5**	**94.1**	**94.3**

Table 6. Ablation study of H2O2Net.

Dataset	Model	Complexity	Prec.	Rec.	F1	FLOPs(G)	$\text{Params}_{decoder}$	Inference Time (1/32)
NYT*	H2O2Net	$\mathcal{O}(L^2 + KL)$	**93.3**	92.3	**92.8**	23.623	1255009	14.9/5.2
	H2O2Net-	$\mathcal{O}(KL)$	89.0	**92.4**	90.7	**0.015**	**73824**	**11.7/2.8**
WebNLG*	H2O2Net	$\mathcal{O}(L^2 + KL)$	**94.5**	94.1	**94.3**	23.713	1707181	14.4/6.0
	H2O2Net-	$\mathcal{O}(KL)$	93.1	**94.2**	93.6	**0.105**	**525996**	**12.3/3.6**

Detailed Results on Complex Scenarios. To verify the ability of our method to handle overlapping patterns and multiple triples, we conduct further experiments on NYT* and WebNLG* datasets. Four powerful models are selected as baselines and the detailed results are shown in Table 3.

It can be observed that the performance of H2O2Net is better than other baselines almost in every subset on both datasets, i.e. 15 of the 18 subsets, especially for most complex scenarios. Compared with PRGC, which is the only model that outperforms H2O2Net on some subsets of NYT*, our proposed model still achieves competitive performance. We attribute the slight performance gap to the fewer number of relations on NYT* than on WebNLG* (24 vs 171) as shown in Table 1 since H2O2Net is a *relation as link* model and the relation plays an important role which means fewer relations may confuse the Linking Strategy. In general, these further experiments adequately prove the effectiveness of the *relation as link* idea and the advantages of our model in complex scenarios.

Results on Different Subtasks. We further explore the performance of H2O2Net on different subtasks, i.e. entity pair extraction and relation classification. The detailed results are reported in Table 5. As previous works [19] pointed out, it can be observed that entity pair recognition and triple formation are two challenging tasks of the joint extraction task.

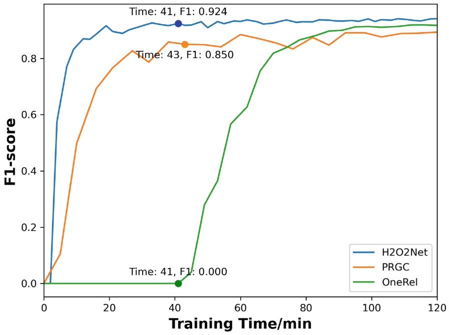
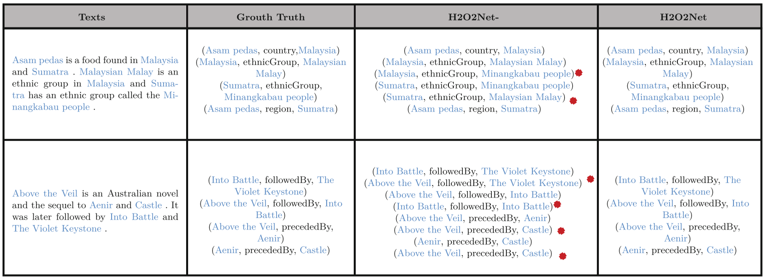


Fig. 7. F1-score in respect to the Training time (ms) on the WebNLG* validation set of different methods.

As shown in Table 5, H2O2Net outperforms all the baselines on most indicators of NYT* and WebNLG*. We attribute the outstanding performance of H2O2Net to its two advantages: First, the relation plays the most important role in a triple for *relation as link* methods, which empowers our model to pay more attention to relation classification than the extraction of other elements. Second, the combination of the Linking Strategy and Link Matrix Module filters out the invalid triples which are extracted by aligning entity-relation pairs. Thus, the performance of subtasks and relational triple extraction is higher than most baselines.



Texts	Grouth Truth	H2O2Net-	H2O2Net
Asam pedas is a food found in Malaysia and Sumatra . Malaysian Malay is an ethnic group in Malaysia and Sumatra has an ethnic group called the Minangkabau people .	(Asam pedas, country,Malaysia) (Malaysia, ethnicGroup, Malaysian Malay) (Sumatra, ethnicGroup, Minangkabau people) (Asam pedas, region, Sumatra)	(Asam pedas, country, Malaysia) (Malaysia, ethnicGroup, Malaysian Malay) (Malaysia, ethnicGroup, Minangkabau people) ✗ (Sumatra, ethnicGroup, Minangkabau people) (Sumatra, ethnicGroup, Malaysian Malay) ✗ (Asam pedas, region, Sumatra)	(Asam pedas, country, Malaysia) (Malaysia, ethnicGroup, Malaysian Malay) (Sumatra, ethnicGroup, Minangkabau people) (Asam pedas, region, Sumatra)
Above the Veil is an Australian novel and the sequel to Aenir and Castle . It was later followed by Into Battle and The Violet Keystone .	(Into Battle, followedBy, The Violet Keystone) (Above the Veil, followedBy, Into Battle) (Above the Veil, precededBy, Aenir) (Aenir, precededBy, Castle)	(Into Battle, followedBy, The Violet Keystone) (Above the Veil, followedBy, The Violet Keystone) ✗ (Above the Veil, followedBy, Into Battle) (Into Battle, followedBy, Into Battle) ✗ (Above the Veil, precededBy, Aenir) (Above the Veil, precededBy, Castle) ✗ (Aenir, precededBy, Castle) (Above the Veil, precededBy, Castle) ✗	(Into Battle, followedBy, The Violet Keystone) (Above the Veil, followedBy, Into Battle) (Above the Veil, precededBy, Aenir) (Aenir, precededBy, Castle)

Fig. 8. Case study for the ablation study of H2O2Net. Examples are from WebNLG and entities are in blue. The red cross marks invalid triples. (Color figure online)

5 Analysis

5.1 Model Efficiency

As shown in Table 4, we evaluate the model efficiency concerning *Complexity*, *FLOPs*, parameters of the decoder ($Params_{decoder}$) and *Inference Time*[2] of

[2] The *FLOPs* and $Params_{decoder}$ are calculated via https://github.com/Lyken17/pytorch-OpCounter.

OneRel, PRGC and H2O2Net on two datasets. All experiments are conducted with the same hardware configuration. The inference time is measured with the batch size of 1 and 32 to verify the single-thread decoding speed and parallel processing capability, respectively. Note that OneRel is restricted to processing one sentence at a time in the official implementation, which is improved to handle data in batch mode. However, it is possible for OneRel to be designed to handle just one sentence at a time so that the inference time of parallel processing increases.

Results show that the single-thread decoding speed and parallel processing decoding speed of H2O2Net are almost the same as PRGC. Note that PRGC decodes triples on the relations detected by its relation prediction module, but H2O2Net considers all relations, increasing the inference time. Compared with PRGC, H2O2Net is lower in *FLOPs* and has 2× fewer parameters, which means H2O2Net achieves better performance with only half the parameters of PRGC. Compared with OneRel, H2O2Net has fewer parameters and the *FLOPs* is even 150 times lower, thus H2O2Net obtains 2× speedup for single-thread decoding and 10× speedup for parallel processing in the inference stage while the F1-score is better.

To further verify the convergence rate of H2O2Net, Fig. 7 shows the F1-score concerning the training time on the WebNLG* validation set of different methods. Results obtained by the official implementation with default configuration prove our convergence rate and efficiency advantages.

5.2 Ablation Study

As shown in Table 6, we conduct an ablation test: H2O2Net- is the model whose Link Matrix Module is removed. Compared with H2O2Net, H2O2Net- also achieves competitive performance, since the number of unreliable triples described in Sect. 3.3 is small and the Entity-Relation Taggers are enough to extract triples excellently. This experiment proves that the alignment between entity pairs is not the critical module for H2O2Net, which is different from *relation as category* methods. It is noteworthy that the original literature indicates the F1-score of PRGC decreases by 12.4 and 23.6 points on the NYT* and WebNLG* datasets, respectively, when the entity pairs alignment module is omitted. In contrast, H2O2Net shows a more modest decrease of only 2.1 and 0.7 points, respectively.

Compared with H2O2Net, the parameters of H2O2Net- are 17× fewer on the NYT* dataset and 3× fewer on the WebNLG* dataset. Moreover, the *FLOPs* is even 1602× lower on the NYT* dataset and 226× lower on the WebNLG* dataset. Note that H2O2Net outperforms PRGC and OneRel in the number of parameters and *FLOPs*. These experimental results indicate that when the computational budget is limited and the precision required is not strict, the Link Matrix Module of H2O2Net can be removed.

Through the case study shown in Fig. 8, it can be observed that H2O2Net- tends to connect all possible entity-relation pairs to construct triples, which results in some invalid triples. That is because the extreme situations described in Fig. 6 appear in the examples of Fig. 8. However, these situations are rare for the complex sentence with more than 3 triples, i.e. only 2 out of 134 in the subsets of WebNLG. Therefore, all experimental results prove that our *relation as link* method are effective.

6 Conclusion

This paper approaches the task of joint relational triple extraction from a novel perspective, introducing an innovative framework that hinges on (entity, relation) pair linking, moving away from the traditional approach of categorizing relations as discrete labels among entity pairs. To our knowledge, H2O2Net is the pioneering model that considers the linking role of relations, mitigating issues such as excessive dependence on entity pair alignment and computational inefficiency. Experimental results on two public datasets demonstrate that our model surpasses all baseline competitors. Subsequent analyses further validate the effectiveness and computational efficiency.

Acknowledgements. This work was supported by the National Key Research and Development Program of China under Grant 2023YFC3304104.

References

1. Bekoulis, G., Deleu, J., Demeester, T., Develder, C.: Joint entity recognition and relation extraction as a multi-head selection problem. Expert Syst. Appl. **114**, 34–45 (2018)
2. Chan, Y.S., Roth, D.: Exploiting syntactico-semantic structures for relation extraction. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 551–560. Association for Computational Linguistics, Portland, Oregon, USA (June 2011). https://aclanthology.org/P11-1056
3. Dai, D., Xiao, X., Lyu, Y., Dou, S., She, Q., Wang, H.: Joint extraction of entities and overlapping relations using position-attentive sequence labeling. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 6300–6308 (2019)
4. Eberts, M., Ulges, A.: Span-based joint entity and relation extraction with transformer pre-training. arXiv preprint arXiv:1909.07755 (2019)
5. Fu, T.J., Li, P.H., Ma, W.Y.: GraphRel: modeling text as relational graphs for joint entity and relation extraction. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 1409–1418. Association for Computational Linguistics, Florence, Italy, July 2019. https://doi.org/10.18653/v1/P19-1136, https://aclanthology.org/P19-1136
6. Gao, C., et al.: Ergm: a multi-stage joint entity and relation extraction with global entity match. Knowl.-Based Syst. **271**, 110550 (2023)
7. Gardent, C., Shimorina, A., Narayan, S., Perez-Beltrachini, L.: Creating training corpora for NLG micro-planning. In: 55th Annual Meeting of the Association for Computational Linguistics (ACL) (2017)

8. Kenton, J.D.M.W.C., Toutanova, L.K.: Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT, pp. 4171–4186 (2019)
9. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
10. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: a lite bert for self-supervised learning of language representations. In: International Conference on Learning Representations (2019)
11. Li, Z., Fu, L., Wang, X., Zhang, H., Zhou, C.: RFBFN: a relation-first blank filling network for joint relational triple extraction. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pp. 10–20. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-srw.2, https://aclanthology.org/2022.acl-srw.2
12. Liu, J., Chen, S., Wang, B., Zhang, J., Li, N., Xu, T.: Attention as relation: learning supervised multi-head self-attention for relation extraction. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 3787–3793 (2021)
13. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
14. Ma, L., Ren, H., Zhang, X.: Effective cascade dual-decoder model for joint entity and relation extraction. arXiv preprint arXiv:2106.14163 (2021)
15. Nayak, T., Ng, H.T.: Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 8528–8535 (2020)
16. Ren, F., et al.: A novel global feature-oriented relational triple extraction model based on table filling. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 2646–2656. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (November 2021). https://doi.org/10.18653/v1/2021.emnlp-main.208, https://aclanthology.org/2021.emnlp-main.208
17. Riedel, S., Yao, L., McCallum, A.: Modeling relations and their mentions without labeled text. In: Balcázar, J.L., Bonchi, F., Gionis, A., Sebag, M. (eds.) ECML PKDD 2010. LNCS (LNAI), vol. 6323, pp. 148–163. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15939-8_10
18. Shang, Y.M., Huang, H., Mao, X.: Onerel: joint entity and relation extraction with one module in one step. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, pp. 11285–11293 (2022)

19. Sui, D., Zeng, X., Chen, Y., Liu, K., Zhao, J.: Joint entity and relation extraction with set prediction networks. IEEE Trans. Neural Netw. Learn. Syst. (2023)
20. Wang, Y., Yu, B., Zhang, Y., Liu, T., Zhu, H., Sun, L.: TPLinker: single-stage joint extraction of entities and relations through token pair linking. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 1572–1582. International Committee on Computational Linguistics, Barcelona, Spain (Online) (December 2020). https://doi.org/10.18653/v1/2020.coling-main.138, https://aclanthology.org/2020.coling-main.138
21. Wei, Z., Su, J., Wang, Y., Tian, Y., Chang, Y.: A novel cascade binary tagging framework for relational triple extraction. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 1476–1488. Association for Computational Linguistics, Online (July 2020). https://doi.org/10.18653/v1/2020.acl-main.136, https://aclanthology.org/2020.acl-main.136
22. Xu, B., et al.: Emrel: joint representation of entities and embedded relations for multi-triple extraction. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 659–665 (2022)
23. Yu, B., et al.: Joint extraction of entities and relations based on a novel decomposition strategy. In: European Conference on Artificial Intelligence (2020)
24. Yuan, Y., Zhou, X., Pan, S., Zhu, Q., Song, Z., Guo, L.: A relation-specific attention network for joint entity and relation extraction. In: IJCAI, vol. 2020, pp. 4054–4060 (2020)
25. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. J. Mach. Learn. Res. **3**(Feb), 1083–1106 (2003)
26. Zeng, D., Zhang, H., Liu, Q.: Copymtl: copy mechanism for joint extraction of entities and relations with multi-task learning. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 9507–9514 (2020)
27. Zeng, X., Zeng, D., He, S., Liu, K., Zhao, J.: Extracting relational facts by an end-to-end neural model with copy mechanism. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 506–514. Association for Computational Linguistics, Melbourne, Australia (July 2018). https://doi.org/10.18653/v1/P18-1047, https://aclanthology.org/P18-1047
28. Zheng, H., et al.: PRGC: potential relation and global correspondence based joint relational triple extraction. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 6225–6235. Association for Computational Linguistics, Online (August 2021). https://doi.org/10.18653/v1/2021.acl-long.486, https://aclanthology.org/2021.acl-long.486
29. Zheng, S., Wang, F., Bao, H., Hao, Y., Zhou, P., Xu, B.: Joint extraction of entities and relations based on a novel tagging scheme. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1227–1236. Association for Computational Linguistics, Vancouver, Canada (July 2017). https://doi.org/10.18653/v1/P17-1113, https://aclanthology.org/P17-1113

30. Zhong, Z., Chen, D.: A frustratingly easy approach for entity and relation extraction. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 50–61. Association for Computational Linguistics, Online (June 2021). https://doi.org/10.18653/v1/2021.naacl-main.5, https://aclanthology.org/2021.naacl-main.5
31. Zhou, G., Su, J., Zhang, J., Zhang, M.: Exploring various knowledge in relation extraction. In: Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), pp. 427–434. Association for Computational Linguistics, Ann Arbor, Michigan (June 2005). https://doi.org/10.3115/1219840.1219893, https://aclanthology.org/P05-1053

BADA-LAT: Efficient Local Attention Transformer for Chinese Named Entity Recognition with Boundary and LLM-Based Data Augmentation

Xiaoping Qiu[1,2,3,4,5,6], Ke Yang[1,4,5](✉), and Shiling Du[1,6]

[1] School of Computing and Artificial Intelligence, Southwest Jiaotong University,
No. 999 Xi'an Road, Chengdu 611756, Sichuan, China

[2] Tangshan Institute of Southwest Jiaotong University,
No. 6 Jingqu Road, Tangshan 063000, Hebei, China

[3] National United Engineering Laboratory of Integrated and Intelligent Transportation, No. 111 North Section 1 Second Ring Road, Chengdu 610031, Sichuan, China

[4] Sichuan Key Laboratory of Manufacturing Industry Chain Collaboration and Information Support Technology, No. 999 Xi'an Road,
Chengdu 610016, Sichuan, China

[5] Engineering Research Center of Sustainable Urban Intelligent Transportation,
No. 999 Xi'an Road, Chengdu 611756, Sichuan, China
a1355041081@gmail.com

[6] Sichuan Institute of Industrial Software Technology, No. 61 Sha'wan Road,
Chengdu 610031, Sichuan, China

Abstract. Progress in Chinese Named Entity Recognition (CNER) has highlighted lexicon-based methods that use word information to boost performance. However, these methods neglect two crucial aspects: the regularity of word boundary and the characteristics of the NER task. To address these shortcomings, we introduce BADA-LAT, a Transformer architecture that incorporates word boundary information and introduce local attention computation. It can enhance entity boundary recognition and concentrate the attention weight of the target token on adjacent characters and matched words. Furthermore, to mitigate the issue of class imbalance, we augment the original training data using large language model (LLM). Our method outperforms other lexicon-based ones, as shown in experiments on four Chinese datasets.

Keywords: Chinese NER · boundary augmentation · local attention · data augmentation

1 Introduction

Named Entity Recognition (NER) is a fundamental step in natural language understanding. It plays a pivotal role in implementing various downstream tasks,

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 383–398, 2025.
https://doi.org/10.1007/978-3-031-78495-8_24

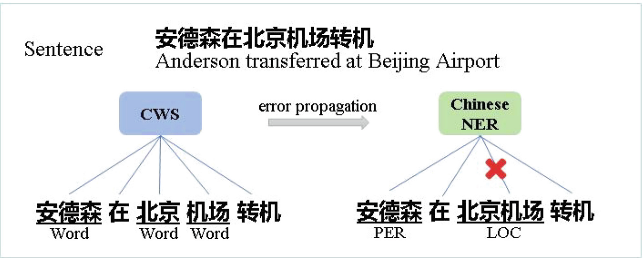


Fig. 1. The results of Chinese Word Segmentation System (CWS) and Named Entity Recognition are compared. CWS incorrectly divides ‘北京机场’ into ‘北京’ and ‘机场’, but this subsequence often represents an entity in the NER task.

including relation extraction [1] and event extraction [2]. The primary focus of NER is to extract crucial information such as persons, locations, and organizations from unstructured text.

However, Chinese NER faces challenges due to the absence of natural word segmentation. An common approach is to first segment sentences using a Chinese Word Segmentation System, then apply sequence labeling. Unfortunately, errors in word segmentation can significantly impact model performance [3], as presented in Fig. 1. Consequently, many researchers have shifted their focus to direct character-level sequence labeling for Chinese NER [4]. These methods have proven to be more effective than word-level labeling. Obviously, character-level sequence labeling for Chinese NER often results in the loss of significant word information. To address this, recent research has proposed lexicon-based model that utilize word information while avoiding segmentation errors. Motivated by this, Zhang and Yang [5] proposed Lattice-LSTM, designed to accommodate lexicon-matched input sequences. However, the complex architecture of Lattice-LSTM makes it challenging to improve.

Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) struggle with modeling long-distance dependency. Due to its excellent global feature extraction and parallel processing, the Transformer model has gained wide usage in various NLP tasks. However, a key insight is that vanilla Transformer inadequately perceive position encoding for critical direction and distance feature [6], which is crucial for handling NER. Recent Transformer model using relative position encoding instead of triangular encoding has demonstrated superior performance on NER task [7]. The Transformer-based FLAT model [8] employs relative position encoding and lexical augmentation, alleviating the limitations of vanilla Transformer position encoding and integrating latent words matched to a lexicon. Thus, FLAT has achieved great success on Chinese NER. However, issue remain with expressing entity boundary information, potentially requiring the introduction of new representational space. Additionally, NER exhibits strong character correlation, making traditional attention computation less adaptable.

Recent works [9] have indicated that incorporating document-level contextual information can further enhance the accuracy of NER. However, acquiring such context is often impractical [10]. It is challenging to find annotators who are well-versed in specific domains. This often meant tremendous efforts from the staff to generate very limited samples. The accuracy and diversity of the new samples remain concerning. Fortunately, large language models like GPT, New Bing, and Claude enable the efficient generation of more diverse and actual samples [11].

In this paper, we propose BADA-LAT to address the aforementioned issues. The key idea is to explicitly express the entity boundary and adaptively calculate the attention score. We design an boundary bias coding space to explicitly represent the boundary deviation term of a candidate entity. To obtain richer boundary information, we assign a unique embedding to each head of the same bias term. Considering the uniqueness of the NER task input sequence, we introduce a local attention mechanism. Specifically, we involve a penalty term that changes linearly with the relative position in the attention calculation. This allows the target token to give a higher attention score to context tokens and related words. Furthermore, given that the number of entity-labeled instances is much less than the number of non-entity-labeled instances in the NER datasets, we leverage the large language model, Bing AI, to augment the data on four datasets. Experimental results show that our method has a clear advantage over FLAT and other lexicon-based models on four Chinese NER datasets.

2 Related Work

2.1 Lexicon-Based NER

Recently, a growing body of research has been dedicated to integrating word information into the NER task elegantly. A pioneering method, Lattice-LSTM [5], improved NER performance by encoding lexicon-matched words while modifying the LSTM structure to accommodate the new input. Gui et al. [12] proposed LR-CNN, which uses convolutional neural network and a rethinking mechanism to encode character sequences and potential words under different windows in parallel. Additionally, Transformer has been leveraged for lexical enhancement given their parallelization and representational capability, such as in PLT [7] and FLAT [8]. As Chinese glyph often provide meaningful representations, Wu et al. [13] proposed MECT, a cross-transformer approach integrating multi-metadata embedding of Chinese character structures to inject greater native language knowledge. Gu et al. [14] introduced RICON-NER, premised on the regularity of Chinese entity mentions.

2.2 Data Augmentation

Data augmentation in Natural Language Processing is a powerful technique for enhancing data representation. At the word level, data augmentation can involve swapping combinations, partial deletion [15] and synonym replacement [16]. Moreover, embedding-space operations are possible but risk introducing antonyms despite similarity [17]. The improved generative adversarial network

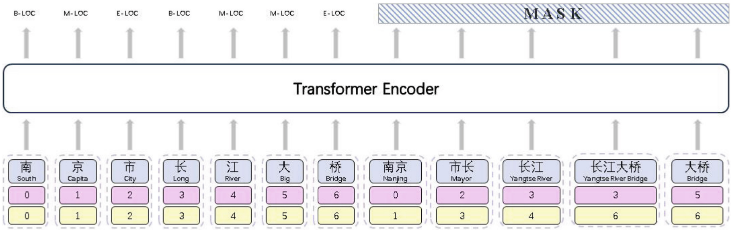


Fig. 2. The input and output of FLAT.

are also used to generate text and have achieved good results. Lin et al. [18] proposed RankGAN which facilitates the generation of richer sentences. Moreover, the diffusion model [19] also provide a unique perspective on data augmentation. Large language models have been leveraged for data augmentation. GPT3Mix [20] leverages large language model to generate realistic text samples from a mixture of real samples. Dai et al. [11] proposed AugGPT which leverages ChatGPT for text classification.

3 Background

The Flat-Lattice Transformer (FLAT) model forms the basis of the proposed approach. Thus, it is necessary to clarify the infrastructure of FLAT model. The following is a brief introduction to the fundamental aspects of the FLAT model.

3.1 Character-Word Fusion Representation Layer

Figure 2 shows the input and output of FLAT. Given a Chinese character sequence $s = [c_1, c_2, ..., c_n] \in \vartheta_c$, where ϑ_c is a character vocabulary. Each character c_i can be represented by a vector:

$$x_i^c = e^c_{(c_i)}, x_i^c \in R^{d_c} \tag{1}$$

where e^c represents the lookup table for character embedding, d_c is the dimension of character embedding. Thus, on the character granularity, the input sequence can be formulated as:

$$X^c = [x_1^c, x_2^c, ..., x_n^c], X \in R^{n \times d_c} \tag{2}$$

After that, FLAT merges the matched word from lexicon and flatten it as a flat counterpart. We use $X^w = [e^w_{(w_1)}, e^w_{(w_2)}, ..., e^w_{(w_m)}], X \in R^{m \times d_w}$ to represent the set of matched word vectors, where w_i and e^w are these words and word embedding lookup, respectively. d_w is the dimension of word embedding. Finally, the representation of the input sequence X is expressed as:

$$X = [W_c X^c; W_w X^w] \tag{3}$$

where W_c and W_w are learnable parameters that are used to align dimension.

Additionally, the relative position coding, R_{i-j}, is calculated as:

$$R_{i-j} = ReLU(W_r(P_{d_{i-j}^{(bb)}} \oplus P_{d_{i-j}^{(be)}} \oplus P_{d_{i-j}^{(eb)}} \oplus P_{d_{i-j}^{(ee)}})), \tag{4}$$

where W_r is a learnable parameter, b and e are the positional index of the first and last character of the candidate entity, respectively. P is obtained as follows:

$$P_{i-j}^{(2k+1)} = cos(\frac{i-j}{10000^{2k/d_{model}}}) \tag{5}$$

$$P_{i-j}^{(2k)} = sin(\frac{i-j}{10000^{2k/d_{model}}}) \tag{6}$$

where $i-j$ represents the relative distance, d_{model} is the hidden size. The self-attention score is calculated using the equation below:

$$\begin{aligned} A_{i,j}^{rel} &= x_i W_Q W_{K,E}^T x_j^T + x_i W_Q W_{K,R}^T R_{i-j}^T \\ &\quad + u W_{K,E}^T x_j^T + v W_{K,R}^T R_{i-j}^T \qquad (7) \\ Att(A,V) &= Softmax(A) X W_V \qquad (8) \\ A_{i,j} &= (\frac{A_{i,j}^{rel}}{\sqrt{d_{head}}}) \qquad (9) \end{aligned}$$

where $W_Q, W_{K,E}, W_{K,R}, W_V, u, v$ are learnable parameters.

3.2 CRF Layer

To model dependency between successive tag, Conditional Random Field (CRF) [21] is often used in NER task. CRF can eliminate irrational decoding rules in the sequence. Given an input sequence $s = [c_1, c_2, ..., c_n]$ and corresponding label sequence $y = [y_1, y_2, ..., y_n]$, CRF aims to maximize the probability of a predicted reasonable continuous tag sequence. The probability is computed as follows:

$$P(y|s) = \frac{exp(\sum_{i=1}^{n} \Phi(y_{t-1}, y_t, x_i))}{\sum_{y' \in Y(s)} exp(\sum_{i=1}^{n} \Phi(y'_{i-1}, y'_i, x_i))} \tag{10}$$

The function $\Phi(y_{t-1}, y_t, x_i)$ calculates the transition score from label y_{t-1} to y_t as well as the emission score for y_t given the input sequence s. $Y_{(s)}$ is the set of all valid label sequences. $P(y|s)$ represents the conditional probability of the label sequence y given s. During decoding, the Viterbi algorithm is used to identify the most probable label path.

4 Approach

This section details our proposed BADA-LAT, as shown in Fig. 3. We first present the boundary augmentation method for candidate entities, then introduce a local attention mechanism to adapt NER task. Lastly, we leverage the large language model to balance datasets.

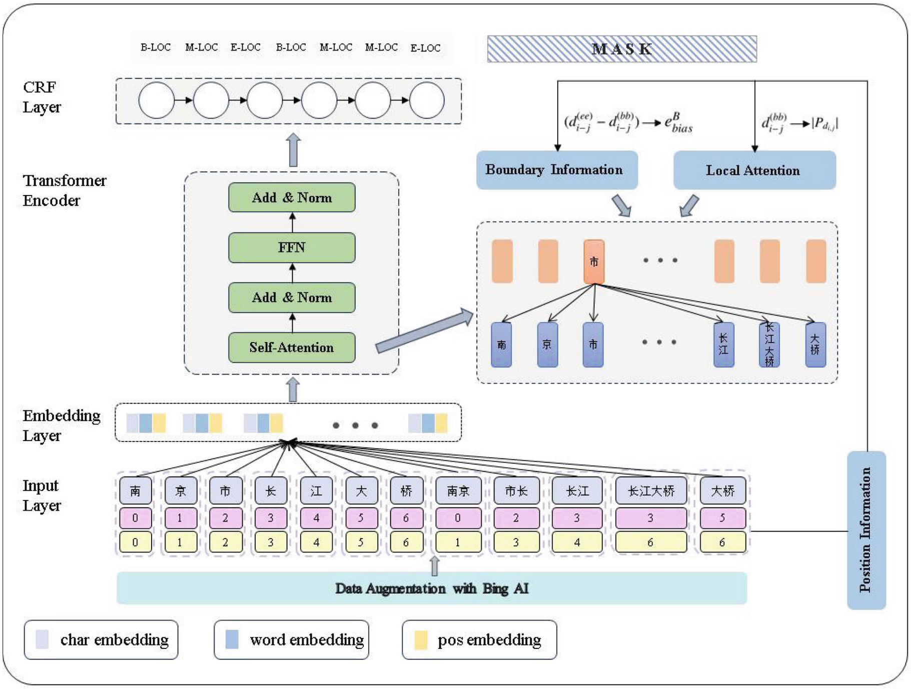


Fig. 3. The overall architecture of BADA-LAT.

4.1 Boundary-Augmentation FLAT-Lattice

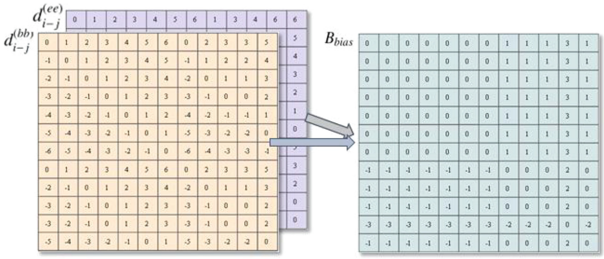
The linear fusion of position is difficult to express boundary information clearly. Therefore, it is necessary to explicitly provide the model with the boundary information of matched word. Specifically, we consider character and word in the lattice as candidate entities, so we get a series of spans. Each span has a corresponding begin and end position. Since the input sequence contains spans of different lengths, we use two relative position to calculate R_{i-j} in a similar way to FLAT:

$$d^{(bb)}_{i-j} = begin[i] - begin[j] \tag{11}$$

$$d^{(ee)}_{i-j} = end[i] - end[j] \tag{12}$$

$$R_{i-j} = ReLU(W_r(P_{d^{(bb)}_{i-j}} \oplus P_{d^{(ee)}_{i-j}})) \tag{13}$$

where i is index of the target token and j is the index of the context token, W_r is a learnable parameter, $begin$ and end are the position index of the first and last character of the candidate entity, respectively. $d^{(bb)}_{i,j}$ is the relative $begin$ position offset of the i-th span with respect to the j-th span, and $d^{(ee)}_{i,j}$ has similar meanings.



0	1	2	3	4	5	6	0	2	3	3	5
-1	0	1	2	3	4	5	-1	1	2	2	4
-2	-1	0	1	2	3	4	-2	0	1	1	3
-3	-2	-1	0	1	2	3	-3	-1	0	0	2
-4	-3	-2	-1	0	1	2	-4	-2	-1	-1	1
-5	-4	-3	-2	-1	0	1	-5	-3	-2	-2	0
-6	-5	-4	-3	-2	-1	0	-6	-4	-3	-3	-1
0	1	2	3	4	5	6	0	2	3	3	5
-2	-1	0	1	2	3	4	-2	0	1	1	3
-3	-2	-1	0	1	2	3	-3	-1	0	0	2
-3	-2	-1	0	1	2	3	-3	-1	0	0	2
-5	-4	-3	-2	-1	0	1	-5	-3	-2	-2	0

0	0	0	0	0	0	0	1	1	1	3	1
0	0	0	0	0	0	0	1	1	1	3	1
0	0	0	0	0	0	0	1	1	1	3	1
0	0	0	0	0	0	0	1	1	1	3	1
0	0	0	0	0	0	0	1	1	1	3	1
0	0	0	0	0	0	0	1	1	1	3	1
0	0	0	0	0	0	0	1	1	1	3	1
-1	-1	-1	-1	-1	-1	-1	0	0	0	2	0
-1	-1	-1	-1	-1	-1	-1	0	0	0	2	0
-1	-1	-1	-1	-1	-1	-1	0	0	0	2	0
-3	-3	-3	-3	-3	-3	-3	-2	-2	-2	0	-2
-1	-1	-1	-1	-1	-1	-1	0	0	0	2	0

Fig. 4. Illustration of boundary offset computation

Indeed, the fusion of position relationship have enhanced its ability to distinguish boundary, albeit to a limited extent. However, there are still a lot of spans with blurred borders due to the lack of representation of boundary information by position coding. As shown in Fig. 4, we can explicitly indicate the boundary bias of spans and encode them in the new encoding space. The boundary bias is calculated as:

$$B_{bias} = (d_{i-j}^{(ee)} - d_{i-j}^{(bb)}) \tag{14}$$

We encode B_{bias} using a new encoding space e_{bias}^{B} that can be learned. Additionally, we assign a different embedding to each head of the same bias term to get richer information. The attention score $A_{i,j}^{rel}$ is calculated as follows:

$$\begin{aligned} A_{i,j}^{rel*} &= (x_i W_Q + v)(W_{K,R}^T R_{i-j}^T + tanh(e_{bias}^{B})) \\ &+ (x_i W_Q + u) W_{K,E}^T x_j^T \end{aligned} \tag{15}$$

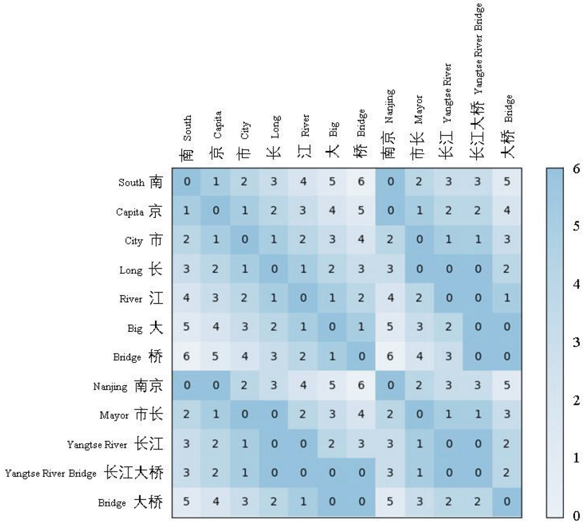
where e_{bias}^{B} are learnable parameter, and propose to activate them using a non-linear function *tanh*. We replace A in the above formula with A^* and then the calculation is the same as FLAT.

4.2 Local Attention Mechanism

In this paper, considering the particularity of NER task including sparse entities and strong correlation of adjacent characters, we introduce a local attention mechanism inspired by ALiBi [22]. This mechanism employs a linear function that adjusts according to changes in relative position, serving as a penalty term. Then we can calculate the attention score as follows:

$$A_{i,j}^{*} = A_{i,j}^{rel} - m \times |P_{d_{i,j}}| \tag{16}$$

where $P_{d_{i,j}}$ is the penalty matrix for attention score, and $|P_{d_{i,j}}|$ represents the absolute value matrix of $P_{d_{i,j}}$. As depicted in Fig. 5, we allocate a penalty value of 0 in $d_{i,j}^{(bb)}$ to word segments linked with the target token. This modified penalty matrix, now referred to as $P_{d_{i,j}}$, allows each token to focus its attention on neighboring characters or related words.



	南 South	京 Capita	市 City	长 Long	江 River	大 Big	桥 Bridge	南京 Nanjing	市长 Mayor	长江 Yangtse River	长江大桥 Yangtse River Bridge	大桥 Bridge
South 南	0	1	2	3	4	5	6	0	2	3	3	5
Capita 京	1	0	1	2	3	4	5	0	1	2	2	4
City 市	2	1	0	1	2	3	4	2	0	1	1	3
Long 长	3	2	1	0	1	2	3	3	0	0	0	2
River 江	4	3	2	1	0	1	2	4	2	0	0	1
Big 大	5	4	3	2	1	0	1	5	3	2	0	0
Bridge 桥	6	5	4	3	2	1	0	6	4	3	0	0
Nanjing 南京	0	0	2	3	4	5	6	0	2	3	3	5
Mayor 市长	2	1	0	0	2	3	4	2	0	1	1	3
Yangtse River 长江	3	2	1	0	0	2	3	3	1	0	0	2
Yangtse River Bridge 长江大桥	3	2	1	0	0	0	0	3	1	0	0	2
Bridge 大桥	5	4	3	2	1	0	0	5	3	2	2	0

Fig. 5. Illustration of the local attention penalty matrix. The penalty factor increases as the relative position increases. Darker color indicate lower penalty term.

We choose the same parameter setting as ALiBi, where m is the set of scalar coefficients fixed for each head. It is a geometric sequence with an initial value of $2^{-\frac{8}{n}}$ and a ratio of $2^{-\frac{8}{n}}$, where n is the number of heads.

The model combining the above methods exhibits minimal changes in time and space complexity. Simultaneously, it gains enhanced boundary resolution capability for candidate entities and a more suitable local attention mechanism for NER task.

4.3 Data Augmentation with BingAI

Since the entity-labled data is much less than the non-entity-labled data, there is a class imbalance problem during training. It is difficult to achieve satisfactory performance through fine-tuning of embedding, such as BERT, because small-scale sample training can easily lead to over-fitting and insufficient generalization ability. Therefore, it requires more correlation contexts during training to improve the performance of NER. We resort to use BingAI for data augmentation. With the help of the ability of LLM, we can easily enhance the context of the sample data to generate more diverse expressions.

Given a set of token sequences $X = [x_1, x_2, ..., x_n]$ and the corresponding label $L = [l_1, l_2, ..., l_n]$, both the predefined prompt and the input sequence are fed into the Bing AI as a question. The prompt can be seen in appendix. The Bing AI performs context enhancement on the input sequence based on the prompt. As shown in Fig. 6, it returns multiple sets of token sequences and character annotation. Since we provide the token labels at the time of input, the labels for the newly generated sequences are nearly error-free, yielding high-quality new

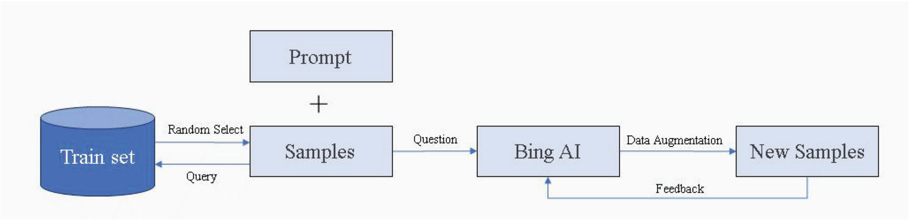


Fig. 6. The data augmentation process with the assistance of Bing AI and an example.

samples. After preprocessing these new samples, we obtain multiple new samples that can be used to expand and enhance the original text from various perspectives. Finally, these new samples are randomly inserted into the training set row by row.

Table 1. Statistics of four benchmarking datasets.

Datasets	Types	Train	Dev	Test
OntoNotes	Sentences	15.72k	4.30k	4.31k
	Entities	4.32k	6.95k	7.70k
MSRA	Sentences	46.36k	–	4.37k
	Entities	74.80k	–	6.20k
Resume	Sentences	3.82k	0.46k	0.48k
	Entities	1.34k	0.16k	0.15k
Weibo	Sentences	1.35k	0.27k	0.27k
	Entities	1.89k	0.39k	0.42k

5 Experiments

5.1 Experimental Settings

To validate the influence of the approach on the model's performance, most of the experiments in this paper follow the FLAT model, such as character and word vector size, dropout, lexicon, etc. We evaluated the proposed method on four Chinese NER datasets, including: OntoNotes 4.0 [23], MSRA [24], Resume [5] and Weibo [25]. Table 1 presents statistic information of these datasets. We use FLAT as the baseline model, and compare our method with other related models.

Table 2. Main results on four Chinese datasets.

Models	OntoNotes 4.0			MSRA			Resume			Weibo		
	P	R	F1	P	R	F1	P	R	F1	NE	NM	Overall
Lattice-LSTM	76.35	71.56	73.88	93.57	92.79	93.18	94.81	94.11	94.46	53.04	62.25	58.79
LR-CNN	76.40	72.60	74.45	94.50	92.93	93.71	95.37	94.84	95.11	57.14	66.67	59.92
CAN-NER	75.05	72.29	73.64	93.53	92.42	92.97	95.05	94.82	94.94	55.38	62.98	59.31
LGN	76.13	73.68	74.89	94.19	92.73	93.46	95.28	95.46	95.37	55.34	64.98	60.21
PLT	76.78	72.54	74.60	94.25	92.30	93.26	95.34	95.46	95.40	53.55	64.90	59.76
SoftLexicon	77.28	74.07	75.64	94.63	92.70	93.66	95.30	95.77	95.53	59.08	62.22	61.42
MECT	77.57	76.27	76.92	94.55	94.09	94.32	96.40	95.39	95.89	61.91	62.51	63.30
FLAT(Baseline)	–	–	76.45	–	–	94.12	–	–	95.45	–	–	60.32
BADA-LAT	78.61	77.32	77.97	95.25	95.52	95.39	96.42	96.10	96.26	61.72	66.22	63.97
With pre-trained Language Model												
BERT	–	–	80.14	–	–	94.95	–	–	95.53	–	–	68.20
BERT + SoftLexicon	–	–	82.81	–	–	95.42	–	–	96.11	–	–	70.50
BERT + MECT	–	–	82.57	–	–	96.24	–	–	95.98	–	–	70.43
RICON-NER	81.95	**84.78**	83.33	95.94	96.33	96.14	–	–	–	–	–	–
BERT + SSMI	**82.46**	84.61	83.52	**96.15**	**96.49**	96.32	**97.48**	**97.18**	97.33	**71.53**	**73.18**	**72.83**
BERT + BADA-LAT	–	–	83.87	–	–	97.33	–	–	**97.48**	–	–	72.26
Roberta + BADA-LAT	–	–	**83.95**	–	–	**97.40**	–	–	97.39	–	–	72.52

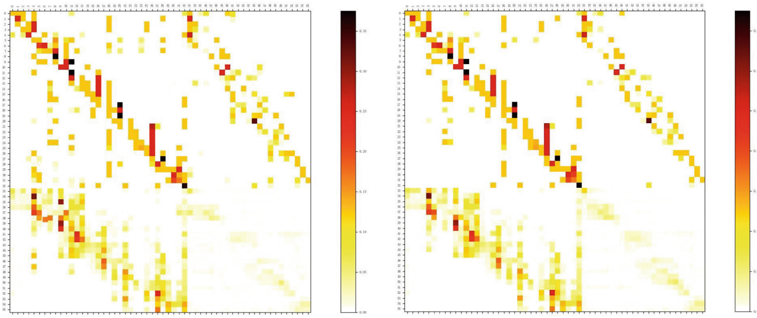
5.2 Comparison with SOTA Methods

Our method is compared with other lexical augmentation models, including Lattice-LSTM [5], LR-CNN [12], CAN-NER [26], LGN [27], PLT [7], SoftLexcion [28], FLAT [8], MECT [13], RICON-NER [14] and SSMI [29].

Table 2 presents the experimental results on the four Chinese datasets. The table is divided into three blocks. The first section lists the experimental results of some classical and recently published methods. The second section compares our method with the baseline method. The third section compares the experimental results integrating the pre-trained language model BERT, released by [30]. In addition, we also use other latest Chinese pre-trained model, roberta [31], for comparison in our model. The bold numbers emphasize the best performance in each evaluation metric. Compared to the recent lexicon-enhanced models based on BERT, our method, with minimal changes to the temporal and spatial complexity of the model, achieves an considerable improvement in the F1 metric on four Chinese datasets, respectively. At the same time, the pre-training model is replaced, and it seems that the performance improvement of the model is not obvious.

5.3 Effectiveness Analysis

In our proposed model, there are two modules directly related to attention computation: boundary enhancement and local attention. Figure 7 contains two heatmaps that show the attention weights of the baseline and our model to the samples, respectively. From the two figures, compared with the baseline, the target token in our model is easier to focus on the word of suitable length and the attention weight is more focused by using the boundary perception and local attention modules than the baseline.



(a) baseline model (b) our model

Fig. 7. Visualization of attention, in which the coordinates 0–32 are used for the characters part and the coordinates 33–55 are for the words part. The two sub-figures show the baseline and our model's attention scores respectively.

5.4 Ablation Study

To better evaluate the contribution of each component within our proposed BADA-LAT architecture, we conduct a ablation study removing individual module and evaluating impact across four datasets as shown in Table 3. Tested elements include boundary bias module, local attention calculation module, and data augmentation with LLM. Experiments show that ablation of these modules almost consistently reduces effectiveness on key NER metrics, affirming the unique role of each component in NER task.

Boundary Bias: Although FLAT used four relative position encoding fusion, it gives the model limited ability to identify candidate entity boundary. However, this attempt seems to be difficult for the model, since boundary information may require different encoding space to be explicitly given. As the Tabel 3 shows, removing the boundary augmentation component, the F1 score of our model has decreased in different degree on the four datasets (0.56% ↓ on OntoNotes4.0, 0.35% ↓ on MSRA, 0.36% ↓ on Resume, 1.07% ↓ on Weibo). Experiments indicate that explicitly giving the boundary bias can enhance the model's perception of boundary.

Local Attention: In the attention score calculation of FLAT, the attention weight between the target token and context tokens are undifferentiated, and the final weight calculation relies on the model's own learning of weights. However, we hold that in NER task, the calculation of attention score should be differentiated, and the attention weight of characters should be more focused on their adjacent characters or related words. As the Tabel 3 shows, replacing our local attention mechanism, the F1 score of our model has decreased in different degree on the three datasets (0.32% ↓ on OntoNotes4.0, 0.23% ↓ on MSRA, 0.30% ↓ on Resume). Experiments indicate that the local attention calculation may be more suitable for NER task.

Data Augmentation with Bing AI: In this study, we removed the data augmentation with Bing AI module as a reference. As the Tabel 3 shows, the F1 score of our model has decreased in different degree on the four datasets (1.34% ↓ on OntoNotes4.0, 1.10% ↓ on MSRA, 1.12% ↓ on Resume, 2.04% ↓ on Weibo). Therefore, data augmentation with LLM can effectively improve the diversity and generalization of training data. The comparison of training sets before and after using LLM data augmentation is counted in appendix.

Table 3. An ablation study of the proposed model.

Models	OntoNotes	MSRA	Resume	Weibo
BADA-LAT	83.87	97.33	97.48	72.26
- Boundary Aug	83.31	96.98	97.12	71.19
- Local Attention	83.55	97.10	97.18	72.53
- Data Aug	82.53	96.23	96.36	70.22

6 Conclusions

In this paper, we introduce the BADA-LAT model, designed based on the regularity of word boundary and the unique characteristics of the named entity recognition (NER) task. The model comprises three submodules. First, a boundary bias term is incorporated into the relative position encoding to capture regularity in candidate entity boundary. Secondly, the attention weight between the target token and its surrounding tokens, as well as associated words, are adjusted to tailor the attention calculation to the needs of NER. Lastly, we leverage prompt-tuning of large language model to perform data augmentation on four Chinese datasets. Experimental results on the four Chinese named entity recognition datasets demonstrate that our method can effectively enhance the performance of Chinese NER.

Acknowledgements. We appreciate the support from the National Key Research and Development Program of China (2023YFB3308602), National Natural Science Foundation of China (72071139), Natural Science Foundation of Hebei Province (G2023105007), and Key Research & Development Program of Sichuan Province (2022YFWZ0007). Their contribution is crucial to the success of our research.

A Appendix

A.1 Prompt of BingAI for Data Augmentation

After repeated fine-tuning, we obtained a most suitable Chinese named entity recognition prompt for large language model text generation. This prompt consist of two parts. The first part is to prompt the large language model BingAI about our task requirements and generation format. The second part is a feedback on the result of the large language model generation, which is used to encourage the large model to continue to generate high-quality text data. The original data of the training sets will be processed into a special format and filled into the prompt. For example, the format of '建设' is '建(O) 设(O)'. The prompt are listed in Table 4.

A.2 Statistics of Data Augmentation

Table 5 show the statistical comparison of the data before and after the augmentation.

Table 4. A prompt for data augmentation in LLM, where "{}" means that sentence will be filled into prompt in a certain format.

prompt	#Generate_Text As a tagger specializing in Chinese named entity recognition data tagging, your task is to generate Chinese text with character tagging based on the reference to "#Examples". The resulting style also should be consistent with the "#Examples" I have provided #Requirements for Text generate You can generate or replace the entities in the "Examples" with other entities of the same type. The new entities should be realistic in the real world and may be not common or famous. The text's content should always be coherent and meaningful. #Examples {} Upon receiving the Examples, your objective is to generate Chinese text with character tagging and should not talk nonsense
feedback	Please repeat all my needs and continue generate more content based on my requirements

Table 5. Statistics before and after data augmentation.

Datasets	Types	Before	After
OntoNotes	Sentences	15.72k	21.64k
	Entities	4.32k	6.27k
MSRA	Sentences	46.36k	57.13k
	Entities	74.80k	87.96k
Resume	Sentences	3.82k	5.42k
	Entities	1.34k	2.21k
Weibo	Sentences	1.35k	2.05k
	Entities	1.89k	3.11k

References

1. Miwa, M., Bansal, M.: End-to-end relation extraction using LSTMs on sequences and tree structures. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, 7–12 August 2016, Berlin, Germany, Volume 1: Long Papers. The Association for Computer Linguistics (2016)
2. Chen, Y., Xu, L., Liu, K., Zeng, D., Zhao, J.: Event extraction via dynamic multi-pooling convolutional neural networks. In: Zong, C., Strube, M. (eds.) Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, pp. 167–176. Association for Computational Linguistics (2015)
3. Peng, N., Dredze, M.: Improving named entity recognition for Chinese social media with word segmentation representation learning. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, 7–12 August 2016, Berlin, Germany, Volume 2: Short Papers, The Association for Computer Linguistics (2016). https://doi.org/10.18653/V1/P16-2025
4. Dong, C., Zhang, J., Zong, C., Hattori, M., Di, H.: Character-based LSTM-CRF with radical-level features for Chinese named entity recognition. In: Lin, C.-Y., Xue, N., Zhao, D., Huang, X., Feng, Y. (eds.) NLPCC 2016. LNCS, vol. 10102, pp. 239–250. Springer, Cham (2016)
5. Zhang, Y., Yang, J.: Chinese NER using lattice LSTM. In: Gurevych, I., Miyao, Y. (eds.) Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, 15–20 July 2018, Volume 1: Long Papers, pp. 1554–1564. Association for Computational Linguistics (2018)
6. Yan, H., Deng, B., Li, X., Qiu, X.: TENER: adapting transformer encoder for named entity recognition. CoRR, abs/1911.04474 (2019)
7. Mengge, X., Yu, B., Liu, T., Zhang, Y., Meng, E., Wang, B.: Porous lattice transformer encoder for Chinese NER. In: Scott, D., Bel, N., Zong, C. (eds.) Proceedings of the 28th International Conference on Computational Linguistics, pp. 3831–3841. International Committee on Computational Linguistics, Barcelona (2020)
8. Li, X., Yan, H., Qiu, X., Huang, X.: FLAT: Chinese NER using flat-lattice transformer. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020, pp. 6836–6842. Association for Computational Linguistics (2020)
9. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics (2019)
10. Wang, X., et al.: Improving named entity recognition by external context retrieving and cooperative learning. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, 1–6 August 2021, p. 1800–1812. Association for Computational Linguistics (2021)
11. Dai, H., et al.: AugGPT: leveraging chatGPT for text data augmentation (2023)

12. Gui, T., Ma, R., Zhang, Q., Zhao, L., Jiang, Y.-G., Huang, X.: CNN-based Chinese NER with lexicon rethinking. In: Kraus, S. (ed.) Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, 10–16 August 2019, pp. 4982–4988. ijcai.org (2019)
13. Wu, S., Song, X., Feng, Z.: MECT: multi-metadata embedding based cross-transformer for Chinese named entity recognition. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 1529–1539. Association for Computational Linguistics (2021)
14. Gu, Y., Qu, X., Wang, Z., Zheng, Y., Huai, B., Yuan, N.J.: Delving deep into regularity: a simple but effective method for Chinese named entity recognition. In: Carpuat, M., de Marneffe, M.-C., Ruíz, I.V.M. (eds.) Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, 10–15 July 2022, pp. 1863–1873. Association for Computational Linguistics (2022)
15. Wei, J.W., Zou, K.: EDA: easy data augmentation techniques for boosting performance on text classification tasks. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019, pp. 6381–6387. Association for Computational Linguistics (2019)
16. Pavlick, E., Rastogi, P., Ganitkevitch, J., Van Durme, B., Callison-Burch, C.: PPDB 2.0: better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In: Zong, C., Strube, M. (eds.) Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Beijing, China, pp. 425–430. Association for Computational Linguistics (2015)
17. Wang, W.Y., Yang, D.: That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using #petpeeve tweets. In: Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., Marton, Y. (eds.) Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, 17–21 September 2015, pp. 2557–2563. The Association for Computational Linguistics (2015)
18. Lin, K., Li, D., He, X., Sun, M.-T., Zhang, Z.: Adversarial ranking for language generation. In: Guyon, I., et al. (eds.) Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA, pp. 3155–3165 (2017)
19. Li, X., Thickstun, J., Gulrajani, I., Liang, P., Hashimoto, T.B.: Diffusion-LM improves controllable text generation. In: NeurIPS (2022)
20. Yoo, K.M., Park, D., Kang, J., Lee, S.-W., Park, W.: GPT3Mix: leveraging large-scale language models for text augmentation. In: Moens, M.-F., Huang, X., Specia, L., Wen-tau Yih, S. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2021, Punta Cana, Dominican Republic, pp. 2225–2239. Association for Computational Linguistics (2021)
21. Lafferty, J.D., McCallum, A., Pereira, F.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: International Conference on Machine Learning (2001)

22. Press, O., Smith, N.A., Lewis, M.: Train short, test long: attention with linear biases enables input length extrapolation (2021)
23. Weischedel, R., et al.: OntoNotes Release 5.0 (2013)
24. Levow, G.-A.: The third international Chinese language processing bakeoff: word segmentation and named entity recognition. In: Ng, H.T., Kwong, O.O.Y. (eds.) Proceedings of the Fifth Workshop on Chinese Language Processing, SIGHAN@COLING/ACL 2006, Sydney, Australia, 22–23 July 2006, pp. 108–117. Association for Computational Linguistics (2006)
25. Peng, N., Dredze, M.: Named entity recognition for Chinese social media with jointly trained embeddings. In: Màrquez, L., Callison-Burch, C., Su, J., Pighin, D., Marton, Y. (eds.) Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, 17–21 September 2015, pp. 548–554. The Association for Computational Linguistics (2015)
26. Zhu, Y., Wang, G.: CAN-NER: convolutional attention network for chinese named entity recognition. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, Volume 1 (Long and Short Papers), pp. 3384–3393. Association for Computational Linguistics (2019)
27. Gui, T., et al.: A lexicon-based graph neural network for Chinese NER. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, 3–7 November 2019, pp. 1040–1050. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1096
28. Ma, R., Peng, M., Zhang, Q., Wei, Z., Huang, X.: Simplify the usage of lexicon in Chinese NER. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, 5–10 July 2020, pp. 5951–5960. Association for Computational Linguistics (2020)
29. Qi, P., Qin, B.: SSMI: semantic similarity and mutual information maximization based enhancement for Chinese NER. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 37, pp. 13474–13482 (2023)
30. Cui, Y., Che, W., Liu, T., Qin, B., Wang, S., Hu, G.: Revisiting pre-trained models for Chinese natural language processing. In: Cohn, T., He, Y., Liu, Y. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 657–668. Association for Computational Linguistics (2020)
31. Zhuang, L., Wayne, L., Ya, S., Jun, Z.: A robustly optimized BERT pre-training approach with post-training. In: Li, S., et al. (eds.) Proceedings of the 20th Chinese National Conference on Computational Linguistics, Huhhot, China, pp. 1218–1227. Chinese Information Processing Society of China (2021)

LTNER: Large Language Model Tagging for Named Entity Recognition with Contextualized Entity Marking

Faren Yan[1,2], Peng Yu[1,2], and Xin Chen[1,2](✉)

[1] Computer Network Information Center, Chinese Academy of Sciences, Beijing, China
{fryan,chx}@cnic.cn, yupeng23@mails.ucas.ac.cn
[2] University of Chinese Academy of Sciences, Beijing, China

Abstract. The use of LLMs for natural language processing has become a popular trend in the past two years, driven by their formidable capacity for context comprehension and learning, which has inspired a wave of research from academics and industry professionals. However, for certain NLP tasks, such as NER, the performance of LLMs still falls short when compared to supervised learning methods. In our research, we developed a NER processing framework called LTNER (Code is available at https://github.com/YFR718/LTNER) that incorporates a revolutionary Contextualized Entity Marking Gen Method. By leveraging the cost-effective GPT-3.5 coupled with context learning that does not require additional training, we significantly improved the accuracy of LLMs in handling NER tasks. The F1 score on the CoNLL03 dataset increased from the initial 85.9% to 91.9%, approaching the performance of supervised fine-tuning. This outcome has led to a deeper understanding of the potential of LLMs.

Keywords: Natural Language Processing · Named Entity Recognition · Large Language Models · Prompt Engineering

1 Introduction

Named Entity Recognition (NER) is an essential component of Natural Language Processing (NLP), playing a vital role in various fields such as text information extraction, information retrieval, and construction of knowledge graphs. The objective of NER is to identify and classify entities in a given text into predefined categories. Although the advent of Transformer models [1] such as Bert [2] and T5 [3] has considerably enhanced the processing capabilities of neural networks on certain datasets, the practical application cost is relatively high due to the substantial data annotation and model training required.

Recently, Large Language Models (LLMs) have emerged as a transformative force in the field of NLP. GPT-3.5 serves as an example of these models' significant abilities in understanding and generating context-rich text. Owing to their numerous params and extensively applied training data [4, 5], these models demonstrate exceptional capacity

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 399–411, 2025.
https://doi.org/10.1007/978-3-031-78495-8_25

for capturing contextual information. Moreover, the convenience of using these models is notable, as they can be easily directed through Prompts to complete an array of tasks, greatly accelerating the deployment of various NLP applications. However, a significant gap exists between LLMs and traditional fine-tuning methods when it comes to NER tasks due to fundamental differences in their output modalities: while NER emphasizes precise annotation [6], GPT models focus on generation; moreover, the issue of hallucinations [7] further compounds the challenge.

In this paper, we introduce LTNER, a method employing contextual marking to leverage the Context-Learning abilities of GPT-3.5 for the improvement of NER tasks without the need for fine-tuning or large-scale training. Experiments conducted on multiple datasets show that our method significantly outperforms existing context NER techniques and closely matches the accuracy of traditional supervised learning methods.

In sum, our contributions are as follows:

1. **An innovative NER method.** We have proposed a simple yet effective Contextual Entity Marking Generation method for large language models (LTNER), which achieves an accuracy close to that of mainstream supervised learning without the necessity for training or fine-tuning.
2. **Robustness of the model.** We have provided ample empirical evidence of LTNER's excellent performance with few contextual examples, limited labeled data, and low cost.
3. **Techniques for optimizing prompt engineering.** We have explored various techniques for optimizing prompt engineering, such as prompt format and role designation, offering valuable insights for future research.

2 LTNER

This chapter provides a detailed exposition of the design principles behind LTNER. Section 2.1 defines the NER task, Sect. 2.2 elaborates on the design of the token generation learning method, and Sect. 2.3 describes the operational flow of the overall system architecture.

2.1 Definition of the NER Task

NER is fundamentally a sequence labeling problem: each token within the text must be assigned a label, indicating whether and how it constitutes a specified entity category. Given a text sequence $X = (x_1, x_2, \ldots, x_n)$, where x_i represents the i-th token in the sequence, the objective of the NER task is to produce a set of entity labels $Y = (y_1, y_2, \ldots, y_n)$, with each y_i selected from a predetermined label set. This label set typically includes $B - Person$ (indicating the beginning of a person's name), $I - Person$ (the continuation of a person name entity), $B - Organization$ (the beginning of an organization name), $I - Organization$ (the continuation of an organization name entity), and other categories such as locations, dates, etc. The O label represents a non-entity.

In traditional supervised learning approaches, a NER model is defined as a conditional probability distribution $P(X \mid Y)$ of the entity label sequence given the text sequence. The training objective of the model is to maximize this conditional probability, which is usually achieved by maximizing the log-likelihood function:

$$\max_{\theta} \sum_{i=1}^{n} logP(y_i \mid X, y_1, y_2, \ldots, y_{i-1}; \theta) \tag{1}$$

Here, θ represents model parameters, and $y_1, y_2, \ldots, y_{i-1}$ denote the labels for the first i − 1 tokens in the sequence. In this way, the NER model learns to allocate the most suitable label for each token, based on the context.

2.2 Contextualized Entity Marking Gen Method

For generative models like GPT, given a prompt sequence $P = (p_1, p_2, \ldots, p_m)$, the goal shifts to maximizing the conditional probability of the desired entity label sequence Y given the text sequence X and the prompt sequence P. This can be expressed using Bayes' theorem as follows:

$$P(Y|X, P; \theta) = \frac{P(Y, X, P|\theta)}{P(X, P; \theta)} \tag{2}$$

where $P(Y, X, P|\theta)$ represents the joint probability of the text sequence X, the prompt sequence P, and the entity label sequence Y under the model parameters θ, and $P(X, P; \theta)$ is the marginal probability of the text sequence X and the prompt sequence P, serving as a normalizing factor to ensure the sum of the probabilities equals 1.

To enhance model performance without altering the model parameters, the primary strategies include:

Designing a Simplified Output Format. To minimize the differences between NER annotation tasks and GPT generation tasks, we have devised a special label marking mechanism. Entities' start and end positions are marked with '##', and the word following '##' represents the entity category. Text not enclosed by '##' does not belong to any entity. This annotation method effectively completes the NER marking mapping while also aligning with GPT's generative pattern, allowing the model to reproduce the original text and insert minimal labels in appropriate places, significantly reducing the interference of extraneous information.

Providing Richer and Utility Context Information. Context learning is a common method to stimulate the learning capability of large models. When a certain amount of annotated data is available, leveraging vector-based retrieval to obtain the most relevant context as the input example and using the annotated results as the output, significantly enhances the large model's ability to perform such tasks.

2.3 Operational Framework

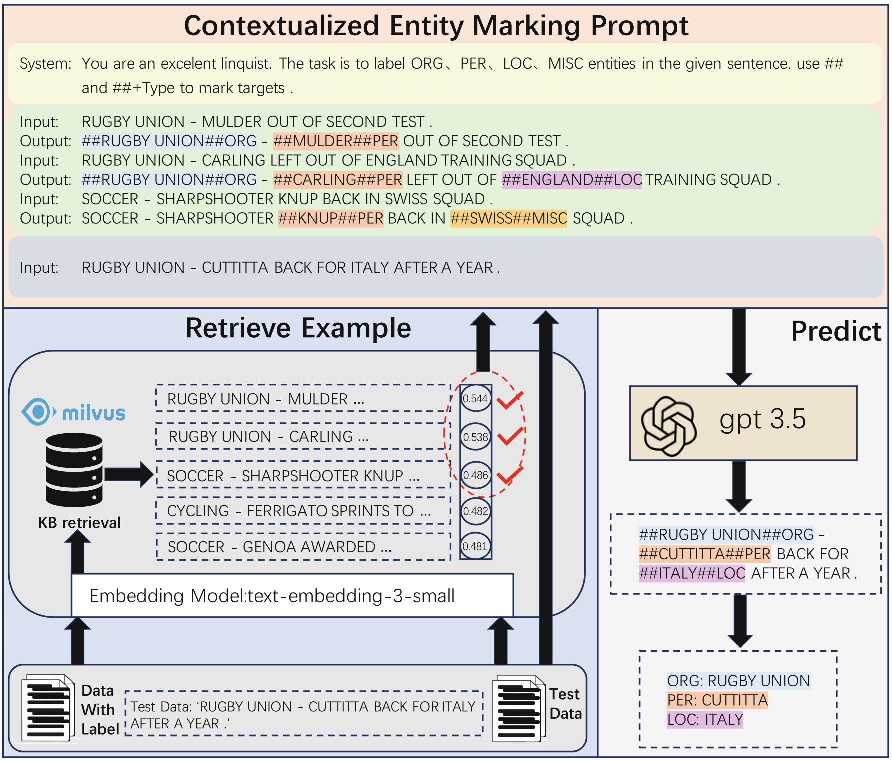

Contextualized Entity Marking Prompt
System: You are an excelent linquist. The task is to label ORG、PER、LOC、MISC entities in the given sentence. use ## and ##+Type to mark targets .
Input: RUGBY UNION - MULDER OUT OF SECOND TEST .
Output: ##RUGBY UNION##ORG - ##MULDER##PER OUT OF SECOND TEST .
Input: RUGBY UNION - CARLING LEFT OUT OF ENGLAND TRAINING SQUAD .
Output: ##RUGBY UNION##ORG - ##CARLING##PER LEFT OUT OF ##ENGLAND##LOC TRAINING SQUAD .
Input: SOCCER - SHARPSHOOTER KNUP BACK IN SWISS SQUAD .
Output: SOCCER - SHARPSHOOTER ##KNUP##PER BACK IN ##SWISS##MISC SQUAD .
Input: RUGBY UNION - CUTTITTA BACK FOR ITALY AFTER A YEAR .
Retrieve Example
milvus
KB retrieval
RUGBY UNION - MULDER ...
RUGBY UNION - CARLING ...
SOCCER - SHARPSHOOTER KNUP ...
CYCLING - FERRIGATO SPRINTS TO ...
SOCCER - GENOA AWARDED ...
0.544
0.538
0.486
0.482
0.481
Embedding Model:text-embedding-3-small
Data With Label
Test Data: 'RUGBY UNION - CUTTITTA BACK FOR ITALY AFTER A YEAR .'
Test Data
Predict
gpt 3.5
##RUGBY UNION##ORG - ##CUTTITTA##PER BACK FOR ##ITALY##LOC AFTER A YEAR .
ORG: RUGBY UNION
PER: CUTTITTA
LOC: ITALY

Fig. 1. The LTNER system framework, comprising three parts: data encoding and vector storage, vector retrieval to construct contextual entity marker learning examples, and result generation and parsing.

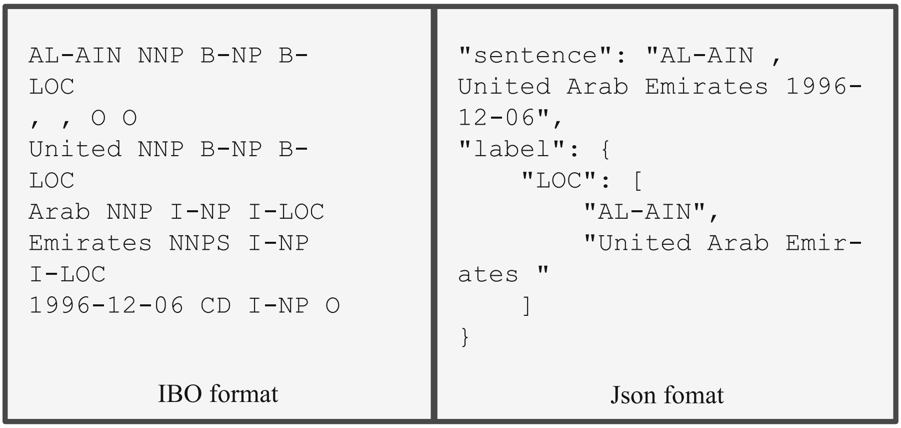
Our system operation is divided into three stages, as shown in Fig. 1. Firstly, we establish a knowledge base by vectorizing the text of training data; vectors, original texts, and annotation results are jointly stored in the vector database. Subsequently, for each piece of test data, the same encoding process is performed to generate the vector, and with these vectors, we retrieve the most similar N pieces of data from the knowledge base. These data are processed according to the previously described annotation method to form context learning examples. Finally, the sentences to be annotated are concatenated with these examples, and the large-scale model generates output results. By parsing these results according to the annotation rules, we derive the predicted output for the NER task.

3 Experiment

3.1 Setup

Datasets. For the empirical data of the NER task, we selected the CoNLL-2003 dataset [8], which is extensively employed in experimental research. The dataset is available in both English and German versions, and for this study, we utilized the English version. It encompasses four categories of entities: person (PER), location (LOC), organizations (ORG), and miscellaneous (MISC). The dataset contains over 20,000 sentences and more than 3,500 entities. To further showcase the method's capacity for generalization, we incorporated an additional dataset, WNUT 2017 [9]. This dataset encompasses six distinct entity types and presents a notable challenge due to its text rife with diverse noise patterns. Its intricate nature renders identification more arduous, rendering it an apt choice for evaluating the efficacy of our approach in managing noisy textual data.

In the data preprocessing phase, we first converted the data from Inside-Outside-Beginning (IOB) format to an easily manipulable raw-text-to-label Json format for subsequent use in Prompt construction.



```
AL-AIN NNP B-NP B-
LOC
, , O O
United NNP B-NP B-
LOC
Arab NNP I-NP I-LOC
Emirates NNPS I-NP
I-LOC
1996-12-06 CD I-NP O
```

IBO format

```
"sentence": "AL-AIN ,
United Arab Emirates 1996-
12-06",
"label": {
    "LOC": [
        "AL-AIN",
        "United Arab Emir-
ates "
    ]
}
```

Json fomat

3.2 Main Results

Our experiment aims to evaluate the effectiveness of contextual learning in the NER task using LTNER. Table 1 presents a comparison of various methods including ours. Notably, the supervised learning and model fine-tuning are represented by InstructUIE [10] and GPT-NER [6]. InstructUIE uses instruction fine-tuning to infuse knowledge, whereas GPT-NER enhances adaptability to NER tasks by training a specialized NER vector encoder. The methods discussed in the middle section are those that do not require fine-tuning, such as CodeIE [11] and Code4UIE [12], which employ a code format output to tightly integrate NER tasks with the powerful code generation capabilities of large models. This class of methods, using content generative models, has shown to be more effective than those using conversational models. The no-fine-tuning version of GPT-NER uses a conventional encoding model to process sentences, outputting labels. However, it extracts only one type of entity at a time and introduces optimization steps like the secondary inspection by large models. The penultimate method employs a Json format output that is easy to parse and well-structured, widely adopted by many mainstream applications today. Finally, our research achieves results through a synthesis of label annotation output and role optimization.

Table 1. Documents the comparison of LTNER with other mainstream methods, including the model name, learning approaches, backbone networks, and the test results of precision, recall, and F1 score.

Method	Paradigm	Backbone	CoNLL 2003			WNUT 2017
			P (%)	R (%)	F1 (%)	F1 (%)
GoLLIE	SFT	Code-LLaMA 34B	–	–	**93.10**	**54.30**
InstructUIE	SFT	Flan-T5-11B	–	–	92.94	–
GPT-NER	SFT + ICL	Text-davinci-003	89.76	92.06	90.91	–
GPT-NER	ICL	Text-davinci-003	83.73	88.07	85.90	40.51
CodeIE	ICL	Code-davinci-002	–	–	82.32	39.67
Code4UIE	ICL	Text-davinci-003	–	–	83.60	41.94
Json	ICL	GPT-3.5-turbo	86.50	83.57	85.01	40.88
LTNER(ours)	ICL	GPT-3.5-turbo	92.86	90.98	**91.91**	49.74

From the data on the right side of the table, it's apparent that prior methods employing ICL average around an 85% F1 score, while those that undergo model fine-tuning hover around 93%. Our context-aware label formatted output method achieves an F1 score of 91.9%, significantly surpassing the existing ICL techniques and closely approaching the fine-tuned models (only a 1% gap). For the WNUT dataset, although the performance gap between LTNER and the fine-tuned model widened (4.6%), LTNER still maintains a significant lead of around 8% over the commonly used ICL method. These findings indicate that our method possesses significant competitive advantages in ICL scenarios.

3.3 Ablation Experiment

Table 2. Records the comparison results of the ablation study, which include variables such as the generation mode, format of the labels, and the role settings for contextual learning. Constants include the number of context samples, the vector retrieval method, and the underlying model, among others.

Pattern	Shots	Tag Combinations	Role Setting	P (%)	R (%)	F1 (%)
Json	30	–	SUA	84.61	82.96	83.78
Tag	30	@@Entry##Label	SUA	91.66	88.33	89.97
Tag	30	##Entry##Label	SUA	91.42	89.48	90.44
Tag	30	@@Entry##Label	AAA	91.53	89.23	90.37
Tag	30	##Entry##Label	AAA	91.42	90.54	90.98

In this section of the ablation study, we meticulously investigated the impact of factors such as generation mode, label formatting, and the role setting in context learning on the performance of the LTNER model. To ensure the accuracy and comparability of our experiments, parameters such as the number of context samples, vector retrieval methods, and the underlying model were held constant, using the CoNLL03 dataset (Table 2).

In comparing generation modes, standard Json format output was juxtaposed with label-based output. With the same 30-shot setting, switching to label-based output resulted in a significant improvement in the F1 score, from 83.7% to 89.9%, indicating a notable enhancement in model performance. Further analysis of various label formats revealed that using '##' as both the beginning and ending symbols for tags, despite a modest decrease in precision (P value), markedly improved the recall rate (R value), and consequently the F1 score. This label format aligns more closely with the training data conventions of LLMs and is better adhered to by the model. For the effects of over ten other label formats, please see Appendix A.1.

For dialogue-based models, assigning appropriate roles when invoking the API is crucial. Typically, we designate the background information as the 'System' role, placed at the beginning of the conversation, and the context examples' queries and answers as the 'User' and 'Assistant' roles respectively, with the text to be identified also portrayed by the 'User' role. This method is referred to as the SUA mode. Experiments have demonstrated that setting all roles to 'Assistant' (thus the AAA mode) also aids in enhancing the recall rate. Such role configurations better suit the nature of NER tasks, given that their inputs and outputs are fixed text and tags, unlike those in a user-machine dialogue setting. For additional role combination effects, see Appendix A.2.

Ultimately, by integrating the double-pound tag notation with the AAA role control strategy, an approximate 1% increment in the F1 score was achieved compared to using the standard label format; as opposed to the conventional Json format output, we observed a significant 7% increase.

The findings underscore the superiority of our approach and provide high-quality empirical evidence for standards set by top-tier conferences in the field of computer science.

4 Analysis

In this section, we will delve into an in-depth analysis of several key factors that affect the performance of the LTNER model. This includes the number of different contextual examples, the quantity of labeled data, and the performance under various levels of expenditure, further exploring the robustness of the LTNER and its advantages in terms of low cost. We conducted these analyses using the CoNLL dataset.

4.1 Different Number of Contextual Example

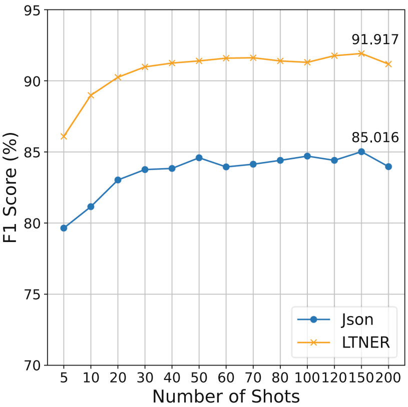


Fig. 2. The Relationship between the Num-ber of Contextual Examples and F1 Score

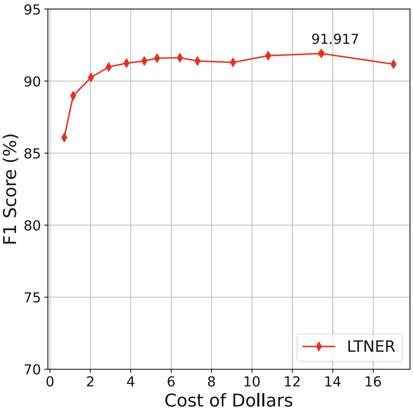


Fig. 3. The Relationship between F1 Scores and Monetary Expenditure

Contextual learning has been proven to be an extremely effective method for improving the performance of large models [5]. The contextual examples of the LTNER are presented based on the marked label format. As shown in Fig. 2, the relationship between the number of examples and the accuracy of the model is apparent. From the figure, we can observe that, starting from 5 samples, the accuracy of the model begins to increase significantly and gradually stabilizes when the number of samples is between 50 and 70. At this point, the accuracies of the Json output format and the LTNER model are 91.61% and 84.59%, respectively. When the number of samples increases to 150, the accuracy of the LTNER model reaches its highest value of 85.08%, surpassing the Json output format, with an accuracy improvement of nearly 7%. As the number of samples continues to rise, there is a slight decline in accuracy, which may be due to the phenomenon of long-context forgetting.

4.2 Different Monetary Expenditure

Currently, large model deployment is more convenient than any deep learning technology before, mainly due to its powerful context comprehension capabilities, eliminating the need for extensive data annotation and model training, as well as separate deployment. The infrastructure for large models is now very mature, exemplified by OpenAI's gpt-3.5-turbo, which has become cost-effective and responsive following several iterations, with a cost of only $0.5 per million input tokens. This section aims to control the number of input context examples and test the accuracy distribution of the dataset at varying costs.

As observed from the results in Fig. 3, there is a significant increase in F1 scores within the cost interval of $0 to $5. With an expenditure of $3, the accuracy reaches 91%, indicating extremely cost-effective performance. At about $5, the F1 score reaches a turning point at 91.62%. Subsequent improvements in accuracy tend to plateau as the expenditure increases, consistent with findings from previous sample size experiments. These results highlight the low-cost advantage of LTNER.

4.3 Different Number of Annotated Data

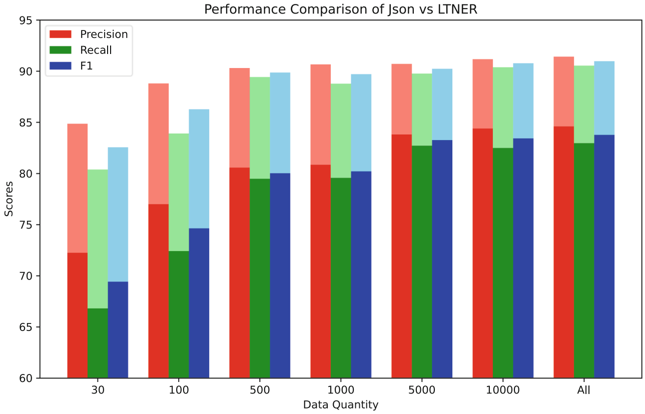


Fig. 4. The Relationship between the Number of Annotated Data and F1 Score (With the Number of Contextual Examples Fixed at 30)

Another significant advantage of employing large language models for NLP tasks is that they do not rely on the vast amounts of annotated data required by traditional supervised learning. A handful of annotated examples can yield quite impressive outcomes. This section of the study examines the results of NER tasks under different volumes of annotated data. To control for variables, we fixed the number of contextual examples at 30 and built a vector retrieval library by randomly selecting a certain amount of annotated data.

The results depicted in Fig. 4 demonstrate that with just 30 annotated examples, the F1 scores for Json and LTNER can reach 79.75% (69.43/83.72) and 90.75% (82.56/90.97) respectively, of the performance attained with full data annotation. Notably, LTNER exhibits markedly superior learning efficiency with the same data amount. When the volume of annotated data increases from 30 to 500, the F1 score rises rapidly, with LTNER achieving 98.7% (89.86/90.97) of the performance of full data annotation with only 1/30 of the annotated data. However, as the volume of data continues to expand, the rate of performance improvement slows down. This experiment vividly illustrates the potent learning capabilities of large models when presented with a small number of samples; a modest amount of data annotation can lead to such high output efficacy.

5 Related Work

Generative Named Entity Recognition. In the field of NLP, NER has become an increasingly studied direction. This method enhances the accuracy of entity recognition by controlling the format of the model's output text and performing multi-step inference. For instance, CodeIE and Code4UIE, leveraging the powerful code generation capabilities of large models, transform the entity recognition task into a code content generation task, thus achieving better structured output. GPT-NER employs a concise tagging and self-checking mechanism, concentrating on extracting entities of a single category. Our research reveals that named entity recognition fundamentally requires strong context support. Spreading out the entity recognition process might lead to misclassifications of the same entity into multiple categories due to a lack of a coherent context.

Multi-stage Named Entity Recognition. Decomposing complex tasks into multiple steps and solving them individually is considered an effective way to enhance the performance of LLMs. ChatIE [13] is a prime example; it deconstructs the information extraction task into a multi-turn dialogue process: first identifying the types of entities to be recognized, then precisely extracting them. The research [14] proposes that by separating content generation from the process of structuring, the model can concentrate on each step independently, thus alleviating the pressure of handling two orthogonal tasks simultaneously. The quality of task prompts is crucial to the model's performance; the zero-sample self-annotation plus checking approach [15] enables the acquisition of a knowledge base in an unsupervised manner, followed by enhancing the inference based on these self-annotated examples. C-ICL [16] utilizes the construction of positive and negative examples for contextual learning demonstrations, thereby strengthening the LLMs' ability to extract entities and relationships.

Fine-Tuned Named Entity Recognition. In the realm of fine-tuned named entity recognition, GPT-NER has developed an entity vector encoder to improve the efficiency and effectiveness of retrieving examples. GoLLIE [17] explores fine-tuning LLMs to meet precise instructional requirements, thus enhancing the zero-shot performance of LLMs in unseen information extraction (IE) tasks. InstructUIE utilizes structured instructions to fine-tune LLMs, which boosts UIE's ability to consistently simulate different IE tasks and capture the dependencies between them. Meanwhile, PaDeLLM-NER [18] significantly accelerates the reasoning speed for NER tasks by parallel decoding all mentions. The study [7] also finds that including negative examples in the training process can significantly improve the model's recognition performance for various tasks.

6 Conclusion

In this study, we introduce an innovative context marking extraction method named LTNER. By adopting a simple tagging generation format, this method substantially enhances the performance of large-scale language models on NER tasks. The outcomes not only surpass existing context-learning based methods but also approach the effects of model fine-tuning. Significantly, our experimental results demonstrate that LTNER achieves effective entity extraction with few samples, sparse annotation data, and at a low cost, thereby offering novel approaches for the rapid deployment and application of NER tasks. We believe that potential future research directions include exploring the performance potential of different large-scale models such as GPT-4, improving LTNER to enhance its recognition capabilities in scenarios involving nested entities, and integrating the automated entity extraction capabilities of large-scale models with a broader range of practical applications to promote the extensive application of NLP technologies.

A. Appendix

A.1. Analyzing the Relationship Between Different Tags and Accuracy

To rapidly ascertain the distinctions among various tags, we conducted experiments using the first 500 entries of the test set, with the number of context examples set at 30 (Table 3).

Table 3. The Relationship Between Tags and NER Performance

Tag Combinations	P (%)	R (%)	F1 (%)
[“##”, “@@”]	91.71	86.58	89.07
[“@@”, “##”]	91.19	86.28	88.66
[“@@”, “##”]	91.72	86.69	89.13
[“##”, “##”]	91.44	89.02	90.22
[“@@”, “@@”]	89.40	88.40	88.90
[“@”, “#”]	91.92	82.20	86.79
[“@”, “@”]	91.43	87.89	89.63
[“#”, “#”]	91.23	86.78	89.47
[“#”, “@”]	92.38	85.15	88.62
[“[”, “]”]	0	0	0
[“”,“”,“”]	0	0	0
[“<”, “>”]	89.11	88.20	88.65

(*continued*)

Table 3. (*continued*)

Tag Combinations	P (%)	R (%)	F1 (%)
[“(”, “)”]	0	0	0
[“+”, “+”]	0	0	0
[“?”, “?”]	0	0	0
[“%”, “%”]	88.31	87.59	87.95
[“”, “”]	86.93	85.34	86.13
[“{”, “}”]	91.40	88.71	90.04
[“{{”, “}}”]	91.12	88.71	89.90
[“[[”, “]]”]	0	0	0
[“”,“”,“”]	0	0	0
[“<< ”, “>>”]	91.67	89.52	90.58
[“((”, “))”]	0	0	0
[“%%”, “%%”]	89.39	89.12	89.26
[“”, “”]	90.34	89.42	89.87

A.2. Analyzing the Relationship Between Different Roles and Accuracy

To rapidly ascertain the distinctions among different roles, we conducted experiments using the entire set of test data, with the number of context examples set at 30 (Table 4).

Table 4. The Relationship Between Role Settings and NER Performance

Role Setting	P (%)	R (%)	F1 (%)
UUU	75.13	82.98	78.86
AAA	85.49	84.31	84.90
SUU	60.77	81.57	69.65
SAA	85.04	83.37	84.19
SUA	84.84	83.36	84.10
AUA	85.46	83.18	84.30
UUA	85.00	83.53	84.26

References

1. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)

2. Kenton, J.D.M.W.C., Toutanova, L.K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT, pp. 4171–4186 (2019)
3. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res. **21**(140), 1–67 (2020)
4. Brown, T.B., et al.: Language models are few-shot learners. arXiv preprint arXiv:2005.14165 (2020)
5. Ou Yang, J., et al.: Training language models to follow instructions with human feedback. arXiv preprint arXiv:2203.02155 (2022)
6. Wang, S., et al.: GPT-ner: named entity recognition via large language models. arXiv preprint arXiv:2304.10428 (2023)
7. Ding, Y., Li, J., Wang, P., Tang, Z., Yan, B., Zhang, M.: Rethinking Negative Instances for Generative Named Entity Recognition. arXiv preprint arXiv:2402.16602 (2024)
8. Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. arXiv preprint cs/0306050 (2003)
9. Derczynski, L., Nichols, E., Van Erp, M., Limsopatham, N.: Results of the WNUT2017 shared task on novel and emerging entity recognition. In: Proceedings of the 3rd Workshop on Noisy User-generated Text, pp. 140–147 (2017)
10. Wang, X., et al.: InstructUIE: multi-task instruction tuning for unified information extraction. arXiv preprint arXiv:2304.08085 (2023)
11. Li, P., et al.: CodeIE: large code generation models are better few-shot information extractors. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 15339–15353 (2023)
12. Guo, Y., et al.: Retrieval-augmented code generation for universal information extraction. arXiv preprint arXiv:2311.02962 (2023)
13. Wei, X., et al.: Zero-shot information extraction via chatting with chatGPT. arXiv preprint arXiv:2302.10205 (2023)
14. Li, Y., Ramprasad, R., Zhang, C.: A simple but effective approach to improve structured language model output for information extraction. arXiv preprint arXiv:2402.13364 (2024)
15. Xie, T., Li, Q., Zhang, Y., Liu, Z., Wang, H.: Self-improving for zero-shot named entity recognition with large language models. arXiv preprint arXiv:2311.08921 (2023)
16. Mo, Y., Yang, J., Liu, J., Zhang, S., Wang, J., Li, Z.: C-ICL: contrastive in-context learning for information extraction. arXiv preprint arXiv:2402.11254 (2024)
17. Sainz, O., García-Ferrero, I., Agerri, R., de Lacalle, O. L., Rigau, G., Agirre, E.: GoLLIE: annotation guidelines improve zero-shot information-extraction. In: The Twelfth International Conference on Learning Representations (2023)
18. Lu, J., Yang, Z., Wang, Y., Liu, X., Huang, C.: PaDeLLM-NER: parallel decoding in large language models for named entity recognition. arXiv preprint arXiv:2402.04838 (2024)

Learning to Generalize Unseen Domains via Multi-source Meta Learning for Text Classification

Yuxuan Hu[1,2], Chenwei Zhang[1,2], Min Yang[4], Xiaodan Liang[1], Chengming Li[2,3](✉), and Xiping Hu[2,3]

[1] Shenzhen MSU-BIT University, Shenzhen, Guangdong, China
{licm,huxp}@smbu.edu.cn
[2] Sun Yat-Sen University, Shenzhen, Guangdong, China
{huyx55,zhangchw7}@mail2.sysu.edu.cn, liangxd9@mail.sysu.edu.cn
[3] Guangdong-Hong Kong-Macao Joint Laboratory for Emotional Intelligence and Pervasive Computing, Shenzhen MSU-BIT University, Shenzhen, Guangdong, China
[4] SIAT, Chinese Academy of Sciences, Shenzhen, Guangdong, China
min.yang@siat.ac.cn

Abstract. With the rapid development of deep learning methods, there have been many breakthroughs in the field of text classification. Models developed for this task have achievedhigh accuracy. However, most of these models are trained using labeled data from seen domains. It is difficult for these models to maintain high accuracy in a new challenging unseen domain, which is directly related to the generalization of the model. In this paper, we study the multi-source Domain Generalization for text classification and propose a framework to use multiple seen domains to train a model that can achieve high accuracy in an unseen domain. Specifically, we propose a multi-source meta-learning Domain Generalization framework to simulate the process of model generalization to an unseen domain, so as to extract sufficient domain-related features. We introduce a memory mechanism to store domain-specific features, which coordinate with the meta-learning framework. Besides, we adopt a novel "jury" mechanism that enables the model to learn sufficient domain-invariant features. Experiments demonstrate that our meta-learning framework can effectively enhance the ability of the model to generalize to an unseen domain and can outperform the state-of-the-art methods on multi-source text classification datasets.

Keywords: Text classification · multi sources · meta-learning · memory

1 Introduction

The text classification of social media is crucial not only for conducting surveys among traditional consumers and companies to gather opinions on respective

This work was supported by Innovation Team Project of Guangdong Province of China (No. 2024KCXTD017).

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 412–428, 2025.
https://doi.org/10.1007/978-3-031-78495-8_26

products or services, but also plays a significant role in national security and public opinion analysis [17,18]. While recent deep learning models of text classification [1–3] demonstrate efficacy in a seen domain (i.e., a domain with labeled data), most of them do not perform well in an unseen domain (i.e., a domain only with unlabeled data). However, in real life, text classification is inevitably used in unseen domains. Text classification can be considered as a domain-dependent task, because a sentence may convey different meanings in various domains. For instance, the term "short" in the context of "short service time" in an electronic review is construed as negative, whereas in a restaurant review, "short" in the context of "short waiting time" is considered positive.

Domain generalization (DG) is a type of transfer learning task. A similar task is Domain Adaptation, which allows access to both source domain and target domain data. Unlike Domain Adaptation, Domain Generalization only permits access to data from the visible domains. The goal of the Domain generalization approach is to address this problem by training a well-generalized model only with labeled data from one or more seen source domains and testing on an unseen domain. Although there has been considerable research on DG in the field of image classification, there have been few studies address in DG of text classification. Most studies in DG of text classification are based on Mixture of Experts (MoE) [4,5]. This approach involves training domain-specific experts and a domain-shared expert independently, followed by their aggregation using a score function. However, the effectiveness of these methods is constrained. In contrast, in real-world scenarios, individuals exhibit a natural ability to swiftly adapt to texts in unknown domains. We posit that the differentiating factor between humans and machines lies in humans' capacity to autonomously categorize domain knowledge into domain-specific and domain-invariant categories and form semantic memory. This ability enables humans to enhance their generalization capacity using prior knowledge. Inspired by this, we contend that storing domain-specific knowledge and domain-invariant knowledge can enhance the DG capabilities of the model.

In this paper, we propose a **M**ulti-source **M**eta-learning framework relying on a "**J**ury" mechanism and **M**emory module (**MMJM**) to facilitate the learning of both domain-invariant and domain-specific features. Specifically, we introduce a meta-learning framework [6] based on the multi-source DG, which simulates how the model generalizes to an unseen domain. We suppose that the meta-learning approach aids the model in differentiating domains and learning the way of classification in an unseen domain by enhancing its ability to capture domain-related features. Additionally, we incorporate a memory mechanism [6] to more effectively capture domain-specific features, leveraging domain information comprehensively while mitigating the risk of unstable optimization. Furthermore, we introduce an innovative "Jury" mechanism [7] to exploit domain-invariant features. This mechanism promotes features from the same class to be closer and features from different classes to be further away.

Our contributions are summarized as follows:

- We propose a novel multi-source meta-learning framework with the memory and "jury" mechanism, which simulates how the model generalizes to an unseen domain.
- We are the first to introduce a memory mechanism into DG for text classification, aimed at learning both domain-specific and domain-invariant features.
- We demonstrate the effectiveness of our proposed framework through extensive experiments and detailed analyses.

2 Related Work

Domain Generalization (DG). In the DG method, the model can access one or more source domains only with labeled data and needs to be tested in the target domain with unlabeled data. Most previous research focuses on extracting domain-invariant representation. The first widely explored method is domain adversarial training. Many studies [4,5,8,9] use this to reduce the divergence between domains. Another method is based on a mixture of experts (MoE). For example, the work [4] propose a set of parallel domain-specific experts to get multiple domain-related classification results and use a distance metric component to compute the mix score to choose the expert. Recently, with the impressive advancements in large language models, many have demonstrated excellent generalization capabilities. Models like ChatGPT (OpenAI), ChatGLM [26,27], Llama [28], Mixtral [29], and others have shown outstanding abilities in domain generalization. These models exhibit excellent classification performance across various domains, demonstrating remarkable domain generalization capabilities.

Meta-learning. The concept of meta-learning is "learning to learn" [10]. Its main idea is to divide the training stage into a meta-train stage and a meta-test stage, using multi-step gradient descent to learn a good initialization. MLDG [11] is the first to incorporate meta-learning into DG, innovatively transforming the meta-train and meta-test process to simulate domain shift situations. This development catalyzes the integration of various meta-learning methods into DG. For example, [12] suggests introducing a regularization function for meta-learning, which is called meta regularizer (MetaReg). What is relevant to us is [6], which introduces a meta-learning framework with a memory module in DG for the Re-Identification task. Different from previous methods, we are the first to introduce the meta-learning method into DG for text classification, combining it with a memory module and a "jury" module.

Contrastive Learning. Contrastive Learning [13] is commonly employed to learn the general features by training models to identify similarities and differences among data points. The representative learning style [14] is to make an anchor closer to a "positive" sample and further from many "negative" samples in the representation space. The work [15] first introduces contrastive learning into text classification. Then, [7] introduces a "jury" mechanism, which can learn domain-invariant features with a memory module. Recently, [16] introduces supervised contrastive learning into DG in text classification to help "learn an ideal joint hypothesis of the source domains". Inspired by these methods, we introduce the "jury" mechanism into DG for text classification.

3 Method

3.1 Problem Definition

The goal of the multi-source DG is to train a model using multiple seen source domains with labeled data, ensuring its effective performance in previously unseen target domains. In this paper, we consider D seen source domains $D_S = \{D_S^d\}_{d=1}^{D}$ as the training set and only one unseen target domain D_T as the test set. The data of each source domain is defined as: $D_S^d = \{(x_{d,i}, y_{d,i})\}_{i=1}^{N_d}$, where N_d is the number of samples in the d-th source domain; $x_{d,i}$ is a sample from the d-th source domain; $y_{d,i}$ is the label of $x_{d,i}$, where $y_{d,i} = \{c\}_{c=1}^{N_C}$, N_C is the number of the class.

3.2 Meta-learning Framework

Following [6], we introduce meta-learning to simulate how the model generalizes to an unseen domain. We divide the training stage into meta-train and meta-test stages. At the beginning of each training epoch, we randomly select one domain's data as the meta-test dataset, and the remaining $D - 1$ domains' data as the meta-train datasets. Our model is shown in Fig. 1.

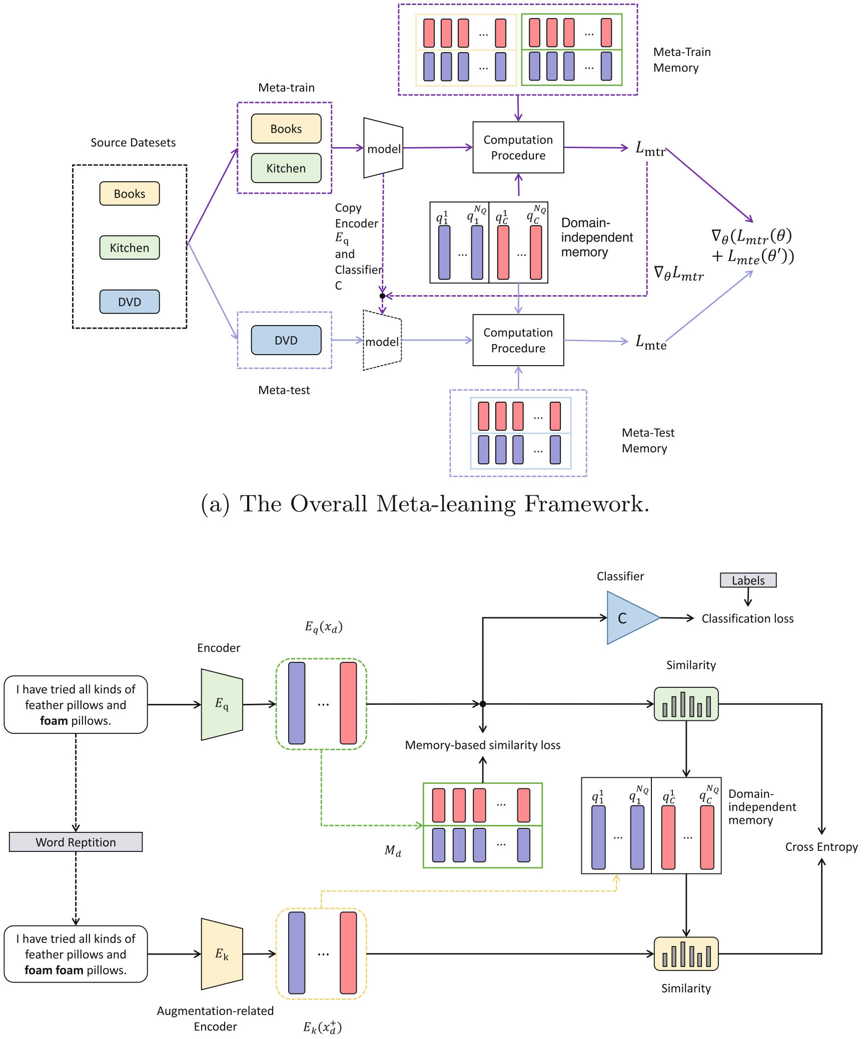
We update the model with meta-train and meta-test stages together. At first, we copy the original model. During the meta-train stage, we calculate the meta-train loss L_{mtr} with the original model. This loss is composed of three parts: the classification loss L_{Class} calculated by the classifier C, the similarity loss L_{Mem} calculated by comparing the features extracted by the encoder with those stored in the memory module, and the L_{Jury} calculated by the "jury" mechanism. Then, we update the copied model with the meta-train loss. During the meta-test stage, meta-test loss L_{mte} is calculated in a similar way as in the meta-train stage. Finally, we update the original model with both the meta-train and meta-test losses. Therefore, the model parameters are updated through the combined meta-train stage and meta-test stages. The final model update formula is shown in Eq. 1:

$$\arg\min_{\theta_{q,C}} L_{mtr}(\theta_{q,C}) + L_{mte}(Adam(\nabla_{\theta_{q,C}} L_{mtr}(\theta_{q,C}), \alpha)) \tag{1}$$

where $\theta_{q,C}$ denotes the parameters of the encoder E_q and the classifier C; L_{mtr} is the meta-train loss; L_{mte} is the meta-test loss; Adam is an optimizer; and α is the inner loop learning rate.

3.3 Memory Module

The memory module is used in conjunction with meta-learning algorithms to store domain-specific features for each domain. We maintain a memory module



(a) The Overall Meta-leaning Framework.

(b) The Detailed Computation Procedure

Fig. 1. The proposed framework MMJM. Figure 1a shows the overall meta-learning framework and the Fig. 1b shows the detailed computation procedure of the meta-train and meta-test stages.

for each domain, denoted as $M = \{M_d\}_{d=1}^{D}$, containing features for each class of that domain, where D is the number of source domains. Each domain's memory module contains N_C feature slots $M_d = \{M_d[c]\}_{c=1}^{N_C}$, where N_C is the number of classes. The dimension of a slot is equal to the dimension of the encoder E_q. We calculate the memory-based similarity loss using classification features encoded by the encoder E_q and the slot $M_d[c]$ of each memory.

Initialization. We initialize all memory modules before the training stage. During initialization, we sequentially initialize all slots in every memory module of all domains in the same way. Specifically, we use the pre-trained encoder to sequentially extract the features of all samples from a class in a source domain. We then initialize the corresponding slot with a domain-specific feature. We calculate the mean of the features of each class and use it as the initialization value for each slot.

Update. After each iteration, we update the slots of each domain sequentially. All slots in all domains are updated in the same way. We update a slot by encoding the features of that class in the current iteration and using a momentum method to update the corresponding class slot, as shown in Eq. 2:

$$M_d[c] = m \cdot M_d[c] + (1 - m) \cdot E[c] \tag{2}$$

where $M_d[c]$ is the slot c of the domain memory d; momentum m is a momentum parameter; $E[c]$ consists of all the features of the c class, defined as: $E[c] = \frac{1}{n}\sum_{i=1}^{n} E(x_{d,i})$, where n is the number of samples of the c class in the d domain for that iteration.

Memory-Based Similarity Loss. We obtain the memory-based similarity loss by calculating the similarity score between the features encoded by the encoder E_q and the slots in the memory module. We calculate the similarity between the feature $E(x_{d,i})$ and the corresponding slot in memory, and normalize these values with softmax. The calculation method is shown in Eq. 3:

$$L_{Mem} = -log\frac{exp((M_d[c])^T E(x_{d,i})/\tau)}{\sum_{c=1}^{C} exp((M_d[c])^T E(x_{d,i})/\tau)} \tag{3}$$

where τ is a temperature parameter.

3.4 "Jury" Mechanism

The domain-invariant features are directly related to the semantic features. We take x^+ which is generated by data augmentation, as the semantically identical sample of input x. We introduce the "jury" mechanism mentioned in [7] to ensure the semantic similarity of each pair, x and x^+. To make the model evolve smoothly and maintain the consistency of representation over time, we construct an augmentation-related encoder, E_k, that updates parameters by momentum.

Word Repetition. Following [15], to change the semantics of the text as little as possible, we choose "word repetition" as our data augmentation method. We randomly repeat some words in a sentence. Given a text x with words $w_1, ..., w_n$ in it, $x = \{w_1, w_2, ..., w_{N_{text}}\}$, where N_{text} is the length of the text. We define the maximum word repetition rate as r. Then, the number of repeated words in a text, k is determined by random sampling within the range $[0, maximum(2,\ \text{int}(r \times N_{text}))]$.This parameter is used to expand the text length during word repetition. In this way, we get the set rep of all repeated words, obtained by uniform sampling. If the words w_1 and w_n are in the set *rep*, the text is converted to $x^+ = \{w_1, w_1, w_2, ..., w_n, w_n, ..., w_{N_{text}}\}$.

Momentum Update. We maintain an augmentation-related encoder E_k. Unlike the encoder E_q, which updates parameters through back propagation, the encoder E_k updates parameters using momentum, as shown in Eq. 4:

$$\theta_k = \lambda \cdot \theta_k + (1 - \lambda) \cdot \theta_q \tag{4}$$

where λ is a momentum parameter; θ_q and θ_k denote the parameters of encoders E_q and E_k, respectively. We believe this update method ensures the smoothness of the encoder E_k updates, reduces differences between the two encoders, and maintains temporal consistency.

"Jury" Mechanism Related Loss. We construct domain-independent memories to store domain-invariant features. We aim for x to have a higher similarity with the semantically identical sample $x^+_{d,i}$ and a lower similarity with other samples. We build a domain-independent memory for each class to store the domain-invariant class features. We define N_C domain-independent memories $Q = (Q_1, ..., Q_c, ..., Q_{N_C})$, where $c \in [1, N_C]$ and N_C is the total number of classes. Each domain-independent memory is structured as a queue of size N_Q: $[q^1_c, ..., q^j_c, ..., q^{N_Q}_c]$, where $j \in [1, N_Q]$, and q^j_c stores the class feature in position j of the Q_c domain-independent memory. We obtain the class features of each instance $x^+_{d,i}$ through the encoder E_k and input them into the corresponding class's domain-independent memory sequentially. We do not distinguish the domain of $x^+_{d,i}$, but store all $x^+_{d,i}$ sequentially into the corresponding class's domain-independent memory. We place each newest feature at the end of the domain-independent memory and delete the oldest feature in the memory.

All features in a domain-independent memory participate in calculating the similarity between $x_{d,i}$ and its augmented sample $x^+_{d,i}$. Both $x_{d,i}$ and $x^+_{d,i}$ compute the cosine similarity with the domain-independent memory corresponding to the current sample's class. We define $S_{d,i} = [s_1, ..., s_j, ...s_{N_Q}]$ as the similarity score between $x_{d,i}$ and all class features stored in the corresponding class's domain-independent memory. We calculate the similarity score between $x_{d,i}$ and the corresponding domain-independent memory using the softmax function, as shown in Eq. 5:

$$s_j = \frac{exp((q^j_c)^T E_q(x_{d,i})/\tau)}{\sum_{j=1}^{N_Q} exp((q^j_c)^T E_q(x_{d,i})/\tau)} \tag{5}$$

Similarly, we define $S^+_{d,i} = [s^+_1, ..., s^+_j, ...s^+_{N_Q}]$ as the similarity score between $x^+_{d,i}$ and all class features stored in the corresponding class's domain-independent memory. The method for calculating the similarity score between $x^+_{d,i}$ and the domain-independent memory is shown in Eq. 6:

$$s^+_j = \frac{exp((q^j_c)^T E_k(x^+_{d,i})/\tau)}{\sum_{j=1}^{N_Q} exp((q^j_c)^T E_k(x^+_{d,i})/\tau)} \tag{6}$$

Then, we penalize cross-entropy loss between the two similarity scores s_j and s_j^+, as shown in Eq. 7:

$$L_{Jury} = -\frac{1}{N_d}\sum_{i=0}^{N_d} s_j^+(x_{d,i}^+)log(s_j(x_{d,i})) \tag{7}$$

where N_d is the number of samples in the d-th source domain.

3.5 Training Procedure

We summarize the overall training process of the model in this section. Before training, we initialize the memory module and the domain-independent memory in the "jury" mechanism. During each iteration, the D datasets are randomly divided into one meta-test dataset and $D-1$ meta-train datasets. Then, the meta-train and meta-test stages work together to optimize the model.

Meta-train. In the meta-train stage, we first extract the same number of training samples $x_{d,i}$ from each meta-train domain. Then we input these samples into encoder E_q to extract features $E_q(x_{d,i})$. The features are subsequently fed into classifier C to calculate classification losses and memory-based similarity loss with the slots in the memory module. Besides, we input augmented samples $x_{d,i}^+$ into encoder E_k to extract augmented features $E_k(x_{d,i}^+)$. The "jury" mechanism-related loss is calculated with features $E_q(x_{d,i})$ and augmented features $E_k(x_{d,i}^+)$. Finally, we update the domain-independent memory built for the "jury" mechanism. The meta-train loss comprises the meta-train classification loss, memory-based similarity loss, and the "jury" mechanism-related loss. The formula is shown in Eq. 8:

$$\begin{aligned} L_{mtr}^d = L_C(X_d;\theta_q;\theta_C) + L_{Mem}(X_d;M_d;\theta_q) + \\ L_{Jury}(X_d;Q;\theta_k) \end{aligned} \tag{8}$$

where θ_q, θ_C, θ_k denote parameters of the encoder E_q, the classifier C, the encoder E_k respectively; X_d and M_d denote the samples and the memory module of domain d; Q denotes the domain-independent memory.

The total meta-train loss is the averaged of the losses from all meta-train domains:

$$L_{mtr} = \frac{1}{D-1}\sum_{d=1}^{D-1} L_{mtr}^d \tag{9}$$

Meta-test. In the meta-test stage, we first copy the encoder E_q and classifier C. Then, we update them with the meta-train loss L_{mtr}. We update the encoder E_k with parameters θ_q', where θ_q' represents the updated parameters of E_q. The meta-test loss is calculated in the same way as the meta-train loss. It is composed of the meta-test classification loss, the memory-based similarity loss, and the "jury" mechanism-related loss. The formula is shown in Eq. 9:

$$L_{mte} = L_C(X_T; \theta'_q; \theta'_C) + L_{Mem}(X_T; M_T; \theta'_q) + L_{Jury}(X_T; Q; \theta'_k) \tag{10}$$

where θ'_q, θ'_C, and θ'_k denote the optimized parameters of the encoder E_q, the classifier C, and the encoder E_k, respectively; X_d and M_d denote the samples and the memory module of d domain; Q denotes the domain-independent memory.

Meta Optimization. Finally, we optimize the model as shown in Eq. 1. Our training procedure is detailed in Algorithm 1:

Algorithm 1: Training Procedure of MMJM

```
Input: $D$ source domains $D_S = \{D_S^d\}_{d=1}^D$.
Initialize: $E_q$ parameterized by $\theta_q$; $C$ parameterized by $\theta_C$;
            $E_k$ parameterized by $\theta_k$; Batch size $B$
            Inner and outer loop learning rates $\alpha$ and $\beta$;
Memory Module Initialization:
foreach $d$ do
    Extract all the samples' features in domain $d$;
    Average features according to classes $C$;
    Initialize memory slots $M_d[C]$ with average values.
end
Domain-invariant memory Initialization:
Random initialize the domain-invariant memory $Q$.
foreach $iter$ do
    Randomly split $D$ into $D_{mtr}$ and $D_{mte}$;
    Meta-train:
    Sample batch $B$ from each domain $X_d = \{x_i\}_{i=1}^B$
    Compute meta-train loss $L_{mtr}$ with Eq. 9;
    Meta-Test:
    Sample batch $B$ from meta-test domain $X_T = \{x_i\}_{i=1}^B$;
    Copy the encoder $E_q$ and classifer $C$ and update $\theta_{q,C}$ :
      $\theta'_{q,C} \leftarrow Adam(\nabla_\theta L_{mtr}, \theta_{q,C}, \alpha)$;
    Compute meta-test loss $L_{mte}$ with Eq. 10;
    Update the domain-invariant memory $Q$;
    Update $\theta_k$ with the Eq. 4
    Update all domain-specific memory $M$;
    Meta Optimization:
    Compute gradient: $g \leftarrow Adam(\nabla_{\theta_{q,C}}(L_{mtr}(\theta_{q,C}) + L_{mte}(\theta'_{q,C}))$
    Update $\theta_{q,C}$ : $\theta_{q,C} \leftarrow Adam(g, \theta_{q,C}, \beta)$
end
```

4 Experimental Setup

4.1 Experimental Data and Evaluation Metrics

To verify the effectiveness of this method, we conduct experiments on two multi-source text classification datasets: the Amazon product review dataset [17] for the multi-source sentiment analysis, and the multi-source rumor detection dataset [18]. The Amazon product review dataset contains 8,000 reviews, evenly distributed across four domains: Books (B), DVDs (D), Kitchens(K), and Electronics (E). Each domain has 1,000 positive reviews and 1,000 negative reviews. The multi-source rumor detection dataset consists of 5,802 annotated tweets from five different events: Charlie Hebdo (CH), Ferguson (F), Germanwings (GW), Ottawa Shooting (OS), and Sydney Siege (SS), labeled as rumors or non-rumors (1,972 rumors, 3,830 non-rumors).

In the experiments, for each dataset, we alternately select one domain as the test set while using the remaining domains as the training set. For evaluation, we use the average accuracy for multi-source sentiment analysis and the average F1 score for multi-source rumor detection, calculated by averaging the results from experiments where each domain is used as the test set.

4.2 Baselines

We compare our method with several state-of-the-art approaches. However, previous methods are mostly based on the Domain Adaptation setting. For fairness, we employ these methods under the Domain Generalization setting.

Basic fine-tunes a model on labeled data from source domains and directly tests it on the target domain. **Gen** refers to [5], which proposes a model composed of several domain-specific CNNs to compute private representations and a shared CNN to compute shared representations, coupled with adversarial training. The MoE model consists of dedicated models for each source domain and a global model trained on labeled data from all source domains. During inference, the ensemble predictions of all models are aggregated. In this context, MoE [9] refers to a MoE model without a pretrained model, while **MoE-Avg** [9] refers to a MoE model with a pretrained model. **PCL** refers to a proxy-based contrastive learning method [19], where the traditional sample-to-sample mechanism is replaced by the proxy-to-sample mechanism. **Intra** [20,21] refers to center loss, which minimizes the distance between each example and its class center. **Adv** refers to the well-studied domain adversarial adaptation method [22], which reverses the gradient calculated by the domain classifier. **Agr-Sum** [23] refers to two gradient agreement strategies based on gradient surgery to reduce the effect of conflicting gradients during domain generalization.

4.3 Implement Details

For all experiments, due to GPU memory constraints, we adapt Distil-Bert-base-uncased [3] and Bert-base-uncased [24] pretrained models as our encoders. The training batch size is set to 8, and we train 15 epochs. We set the token number of samples to 512. To optimize our model, we use the Adam optimizer with a weight decay of 5×10^{-4} to optimize the encoder E_q. The inner loop learning rate α and the outer loop learning rate β start at 1×10^{-6} and are increased to 1×10^{-5} in the first epoch. For the memory module, the momentum coefficient m is set to 0.2 and the temperature factor τ is set to 0.05. For the "jury" mechanism, the momentum coefficient λ is set to 0.999, and the size of the domain-independent memory is set to 64×768.

4.4 Experimental Results

The experimental results of text classification are shown in Table 1. First, in the multi-source sentiment analysis and rumor detection, our method achieves the best performance, demonstrating its effectiveness for DG in the text classification. Second, compared to the meta-learning baseline MLDG and the contrastive learning baseline SCL, our method achieves higher accuracy. This success highlights the advantage of capturing both domain-invariant and domain-specific fea-

Table 1. Experiments Results Compared with State-of-the-art Approaches.

Method	D	B	E	K	Avg Acc	CH	F	GW	OS	S	Avg F1
Gen	77.9	77.1	80.9	80.9	79.20	–	–	–	–	–	–
MoE	87.7	87.9	89.5	90.5	88.90	–	–	–	–	–	–
DistilBert											
MoE-Avg	88.9	90.0	90.6	90.4	89.98	67.9	45.4	74.5	62.6	64.7	63.02
SCL	**90.1**	90.0	90.3	**90.8**	90.30	**68.1**	44.5	75.4	66.5	65.2	63.94
Basic	89.1	89.8	90.1	89.3	89.58	66.1	44.7	71.9	61.0	63.3	61.40
MLDG	89.5	90.3	**90.8**	90.7	90.33	66.1	52	**79.8**	69.9	63.1	66.18
PCL	89.2	89.8	90.3	90.5	89.95	60.4	50.1	75.7	70.9	64.5	64.32
Intra	88.5	89.8	90.1	89.2	89.40	64.1	42.9	70.8	61.8	62.4	60.40
Adv	88.4	89.0	89.6	90.0	89.25	64.8	42.2	65.9	61.4	62.8	59.42
Agr-Sum	88.8	89.1	90	90.3	89.55	67.5	52.0	76.9	69.0	64.3	65.94
MMJM	89.8	**90.5**	**90.8**	**90.8**	**90.48**	66.3	**52.3**	77.1	**73.4**	**72.7**	**68.36**
Bert											
MoE-Avg	90.4	91.4	91.5	92.3	91.4	67.7	46.7	**80.8**	53.7	60.5	61.88
Basic	90.5	91.2	92.2	91.9	91.45	66.6	46.0	73.7	69.2	61.8	63.46
MLDG	91.1	91.9	91.8	92.5	91.83	67.8	50.3	77.3	71.3	60.5	65.44
PCL	89.5	91.4	**92.4**	91.8	91.28	64.1	**53.9**	70.1	70.9	66.2	65.04
Intra	90.4	91.2	91.3	92.1	91.25	63.4	47.9	71.0	67.6	57.7	61.52
Adv	91.1	91.2	91.1	92.0	91.35	66.1	44.4	69.0	70.3	62.2	62.40
Agr-Sum	90.2	91.1	91.2	91.8	91.08	67.9	53.4	77.3	70.5	57.6	65.34
MMJM	**91.2**	**91.7**	91.9	**92.7**	**91.88**	**68.8**	52.6	78.3	**73.0**	**70.6**	**68.66**

tures. Third, our method does not reach peak performance in every domain. We attribute this to the ambiguous characteristics of data in some areas. Nonetheless, our strong performance across domains, as evidenced by average scores, highlights the method's adaptability in diverse settings.

4.5 Ablation Studies

To further analyze the effectiveness of each part of our model, we conduct ablation studies. The results are shown in Table 2, where "Meta" indicates training with the meta-learning framework, "Mem" indicates training with the memory module, "Jury" indicates training with the "jury" mechanism, "SA" refers the multi-source sentiment Analysis, and "RD" refers the multi-source rumor detection. All ablation studies are based on distil-Bert.

Effectiveness of Meta-learning. We conduct the ablation study to investigate the proposed meta-learning strategy. The model trained with the proposed meta-learning strategy could improve the results. For sentiment analysis, the model trained with a meta-learning strategy increases the basic baseline by 1.02% in the average classification accuracy. For rumor detection, the average F1 score increases by 7.6%. We believe that the implementation of meta-train and meta-test processes within the meta-learning framework could help the model adapt to the training of multi-source domains and learn domain-related features. This approach potentially reduces the risk of the model overfitting to domain biases, thereby enhancing the model's performance in encountering unseen domains. However, the introduction of the meta-learning approach consumes substantial computational resources and memory. We believe this is due to the characteristics of the meta-learning method, which combines data from multiple source domains and updates parameters in two stages.

Table 2. Ablation Experiments

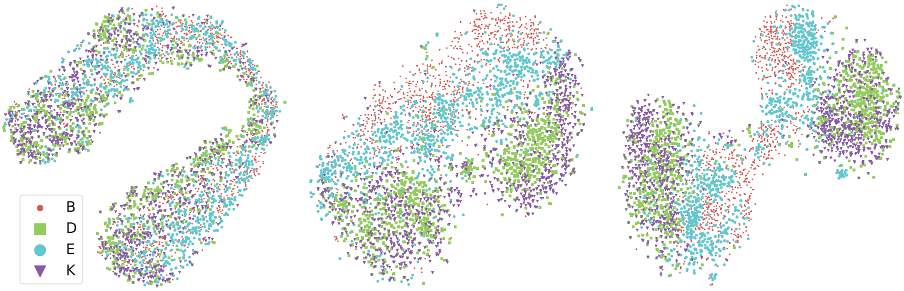
Meta	Mem	Jury	SA Avg Acc	RD Avg F1	RAM usage	GPU usage
×	×	×	89.58	61.40	2711.71 MB	1058.31 MB
✓	×	×	**90.6**	69.00	2871.52 MB	1816.93 MB
×	×	✓	90.28	66.96	2764.84 MB	1312.36 MB
✓	✓	×	90.38	68.30	2941.57 MB	1817.65 MB
✓	×	✓	90.33	**69.32**	2894.88 MB	2072.82 MB
✓	✓	✓	90.43	68.36	**3031.73 MB**	**2073.60 MB**

Effectiveness of the "Jury" Mechanism. The "jury" mechanism related loss could increase the classification accuracy. For sentiment analysis, compared with the Distil-Bert baseline, the model trained with the "jury" mechanism could achieve an average accuracy increase of 0.7%. For rumor detection, the average F1 score increases by 5.56%. We believe that this is because the model is trained with a large number of data-augmented examples, which could help the model learn to reduce the domain divergence between different domains. What's more, it's obvious that the average F1 score increases by 0.32% for rumor detection, but the average accuracy reduces by 0.27% for sentiment analysis when combining the meta-learning method and the "jury" mechanism. We think that this observation may be attributed to the more apparent similarities among data from different domains in sentiment analysis. Additionally, we find the 'jury' mechanism demands relatively fewer computational resources and memory. It is a low-cost method that can enhance the model's generalization capabilities.

Effectiveness of Memory Module. Contrary to expectations, the meta-learning framework with the memory module can't further increase the accuracy of classification compared to using meta-learning alone. However, compared to the model without the memory module, MMJM could increase the average sentiment analysis accuracy by 1%. We believe that the memory module is helpful when combined with the domain-invariant features. Additionally, the memory module requires minimal computational resources and relatively low memory. We believe that the introduction of the memory module, despite its low cost, can significantly enhance the generalization capability of the entire framework. Thus, the inclusion of the memory module is highly worthwhile.

4.6 Visualization

To better understand the effectiveness of our method, we provide the t-SNE visualizations [25] of our MMJM and some baselines to intuitively assess the performance of our model on domain discrepancy, as illustrated in Fig. 2. We observe that features from the source and target domains of MMJM are much more compact, which indicates our framework can learn more domain-specific and domain-invariant features.



(a) Distil-Bert (b) PCL (c) MMJM

Fig. 2. t-SNE visualization of the embeddings from Distil-Bert. We choose models with source domains D, K, E and target domain B and sample 1,000 examples for each domain.

4.7 Comparison with Large Language Model

To ensure the comprehensiveness of our experiments, we also explored the performance of large language models on the datasets. The experimental results are shown in Table 3. It can be seen that on the sentiment classification dataset, ChatGPT[1] performs well, with an average classification accuracy 2.1% higher than MMJM. This indicates that large language models have strong sentiment perception abilities and can achieve good results across different domains, demonstrating strong generalization capabilities. However, on the rumor dataset, its performance is poor, which we believe is due to the lack of relevant knowledge about rumor detection in the language model. Therefore, we believe our model has research value, as it uses only 1% of the parameters of ChatGPT yet achieves good classification performance, significantly saving training resources and time costs for training large models. Additionally, it can still achieve good results through training on specific topics.

4.8 Case Study

We present a case study to intuitively understand our framework mechanism, as shown in Table 4. The encoder for both models is distil-Bert. P and N respectively denote the positive and negative predictions. The symbol ✘ indicates a wrong prediction. Cases from four domains B, D, E, and K. The second sentence shows that both the basic baseline and our model MMJM can make the correct prediction. We believe that is because the sentence is relatively simple making it easier to identify key information such as "books" and "stupid". However, the performances of the two models in the third sentence are different. That demonstrates that our MMJM model can address complicated and informal sentences, but the basic model focuses on the sentiment words "kind" and "clear". What's more, we find both our model and the basic model would make the wrong answer when the class-related information or sentiment expression is unclear, like the first and the fourth sentences. Specifically, the fourth sentence contains class-related words like "rip" and "hard", which leads to the mistake, while the first sentence has an unclear sentiment expression. ALL in all, our method makes the true prediction most of the time, so we make the conclusion that our model MMJM can capture the domain-related and class-related information, though there is still room for improvement.

Table 3. Experiments Results Compared with ChatGPT.

Method	D	B	E	K	Avg Acc	CH	F	GW	OS	S	Avg F1
ChatGPT	**93.1**	**93.2**	**94.4**	**95.2**	**93.98**	40.9	36.5	52.0	39.3	48.9	43.52
MMJM	91.2	91.7	91.9	92.7	91.88	**68.8**	**52.6**	**78.3**	**73.0**	**70.6**	**68.66**

[1] gpt-3.5-turbo.

Table 4. Case study between the proposed framework and the basic model.

Text	Basic	MMJM
I saw the scene, where they have lissa chained to the pool table and gagged in the basement. I didn't understand most of the movie. I bet kim possible, Ron Stoppabl, and rufus can deal with them.(D)	N(✘)	N(✘)
I really feel stupid about ordering this book. Why did I do it?. The story is that stacey is moving back to new york.(B)	N	N
Very clear image! Probably the best that I saw on this kind devices! No software availble. Device is useless if you do not have windows media os. Absolutely no linux support. Good hardware... but useless... you have to buy software for $70- 120 to be able use it...(E)	P(✘)	N
It really does make a difference when some of the chlorine is filtered out of your coffee (and even more so in your bourbon) water, but these filters are a rip off price-wise. It's hard to believe braun wouldn't make money selling them at a third of the price.(K)	(✘)	N(✘)

5 Conclusion

In this paper, we propose a multi-source meta-learning framework for DG in text classification. The meta-learning strategy simulates how the model generalizes to an unseen domain. Additionally, we incorporate a memory-based module and the "jury" mechanism to extract domain-invariant features and domain-specific features, further enhancing the model's performance.

References

1. Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R.R., Le, Q.V.: XLNet: generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems 32 (2019)
2. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
3. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
4. Guo, J., Shah, D.J., Barzilay, R.: Multi-source domain adaptation with mixture of experts. arXiv preprint arXiv:1809.02256 (2018)
5. Li, Y., Baldwin, T., Cohn, T.: What's in a domain? Learning domain-robust text representations using adversarial training. arXiv preprint arXiv:1805.06088 (2018)
6. Zhao, Y., et al.: Learning to generalize unseen domains via memory-based multi-source meta-learning for person re-identification. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6277–6286 (2021)
7. Chen, Y., Wang, Y., Pan, Y., Yao, T., Tian, X., Mei, T.: A style and semantic memory mechanism for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9164–9173 (2021)
8. Chen, X., Cardie, C.: Multinomial adversarial networks for multi-domain text classification. arXiv preprint arXiv:1802.05694 (2018)

9. Wright, D., Augenstein, I.: Transformer based multi-source domain adaptation. arXiv preprint arXiv:2009.07806 (2020)
10. Thrun, S., Pratt, L.: Learning to learn: introduction and overview. In: Learning to Learn, pp. 3–17. Springer (1998)
11. Li, D., Yang, Y., Song, Y.-Z., Hospedales, T.: Learning to generalize: meta-learning for domain generalization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)
12. Balaji, Y., Sankaranarayanan, S., Chellappa, R.: MetaReg: towards domain generalization using meta-regularization. In: Advances in Neural Information Processing Systems 31 (2018)
13. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), vol. 2, pp. 1735–1742. IEEE (2006)
14. van den Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 (2018)
15. Wu, X., Gao, C., Zang, L., Han, J., Wang, Z., Hu, S.: ESimCSE: enhanced sample building method for contrastive learning of unsupervised sentence embedding. arXiv preprint arXiv:2109.04380 (2021)
16. Tan, Q., He, R., Bing, L., Ng, H.T.: Domain generalization for text classification with memory-based supervised contrastive learning. In: Proceedings of the 29th International Conference on Computational Linguistics, pp. 6916–6926 (2022)
17. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: domain adaptation for sentiment classification. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pp. 440–447 (2007)
18. Zubiaga, A., Liakata, M., Procter, R., Hoi, G.W.S., Tolmie, P.: Analysing how people orient to and spread rumours in social media by looking at conversational threads. PLoS ONE **11**(3), e0150989 (2016)
19. Yao, X., et al.: PCL: proxy-based contrastive learning for domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7097–7107 (2022)
20. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: European Conference on Computer Vision, pp. 499–515. Springer (2016)
21. Ye, H., Tan, Q., He, R., Li, J., Ng, H.T., Bing, L.: Feature adaptation of pre-trained language models across languages and domains with robust self-training. arXiv preprint arXiv:2009.11538 (2020)
22. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(1) (2016). ISSN: 2096-2030
23. Mansilla, L., Echeveste, R., Milone, D.H., Ferrante, E.: Domain generalization via gradient surgery. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6630–6638 (2021)
24. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
25. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(11) (2008)
26. Zeng, A., et al.: GLM-130B: an open bilingual pre-trained model. arXiv preprint arXiv:2210.02414 (2022)
27. Du, Z., et al.: GLM: general language model pretraining with autoregressive blank infilling. arXiv preprint arXiv:2103.10360 (2021)

28. Touvron, H., et al.: LLaMA: open and efficient foundation language models. arXiv preprint arXiv:2302.13971 (2023)
29. Jiang, A.Q., et al.: Mixtral of experts. arXiv preprint arXiv:2401.04088 (2024)

Enhancing Bengali Text-to-Speech Synthesis Through Transformer-Driven Text Normalization

Krishnendu Ghosh[1](✉), Munmun Patra[2], and Eshita Dey[2]

[1] Indian Institute of Information Technology Dharwad, Dharwad, Karnataka, India
krishnendu@iiitdwd.ac.in

[2] Techno International New Town, Kolkata, West Bengal, India
munmunpatra162@gmail.com, deyeshita03@gmail.com

Abstract. This paper presents a transformer-driven approach for non-standard word (NSW) normalization in Bengali text-to-speech synthesis (TTS) systems. Our text normalization (TN) approach is realized over three modules: pre-processing, NSW classification, and token-to-word expansion. The pre-processing module tokenizes the input sentences into words. The non-standard words are classified based on their format and other discriminative features in the next module. Apart from our proposed transformer-based approach, the present study explored four baseline TN approaches: rule-based, Conditional Random Field (CRF)-based, Long Short-Term Memory (LSTM)-based, and biLSTM-based. Based on the predicted types of the NSWs, standard words (SWs) are generated through token-to-word expansion using hand-crafted expansion rules. The performance of these five TN approaches is assessed on a corpus of around 6,000 NSWs, consisting of 6 initial and 14 final NSW types. The performances of these approaches are assessed using measures representing (i) direct accuracy and (ii) enhancements in text-to-speech synthesis systems. Based on direct accuracy, our proposed approach predicted correct NSW types for about 92% of cases. This performance is around 2% better than the performance of a rule-based approach. On the other hand, application-based measures guarantee that the generated utterances (i) sound natural and (ii) improve the quality of speech synthesis systems.

Keywords: Text-to-speech synthesis · Bengali · Text normalization · Transformer · Non-standard words

1 Introduction

A text-to-speech synthesis (TTS) system is the process of artificially producing human speech by transforming texts into corresponding speech waveforms [7]. TTS systems are screen-reading tools for individuals with visual impairments, announcement systems in railway and flight schedules, and customer service

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 429–444, 2025.
https://doi.org/10.1007/978-3-031-78495-8_27

applications in call centres [9]. The significance of TTS extends far beyond mere convenience, impacting various aspects of accessibility, human-computer interaction, and entertainment. One of the critical challenges in developing good quality TTS systems lies in the inherent variations and inconsistencies present in written texts [10]. Written texts often deviate from spoken language conventions and exhibit inconsistent pronunciations when appearing in diverse contexts. Due to their diverse and ambiguous formats, textual contents comprise a variety of non-standard words (NSWs) and, therefore, pose challenges for TTS systems [14]. Text normalization bridges this division between written and spoken forms and, therefore, helps in generating natural and comprehensible speech in text-to-speech synthesis (TTS) systems, as its precision and phonetic smoothness impact the quality of automatic speech recognition (ASR) systems [8,13].

The current work describes the types and occurrences of various non-standard words (NSWs) within the Bengali text corpus. Non-standard words are those whose pronunciation deviates from their written form in scripts. Examples encompass numbers (year, time, ordinal, cardinal, floating point), abbreviations, acronyms, currency, dates, and URLs. Typographic irregularities such as digit sequences, acronyms, and letter sequences are predominantly categorized as NSWs [12]. Text normalization detects and converts these non-standard words into their corresponding standard word (SW) counterparts. The comprehensive block diagram depicted in Fig. 1 illustrates the architecture of a text normalization module utilized in TTS systems.

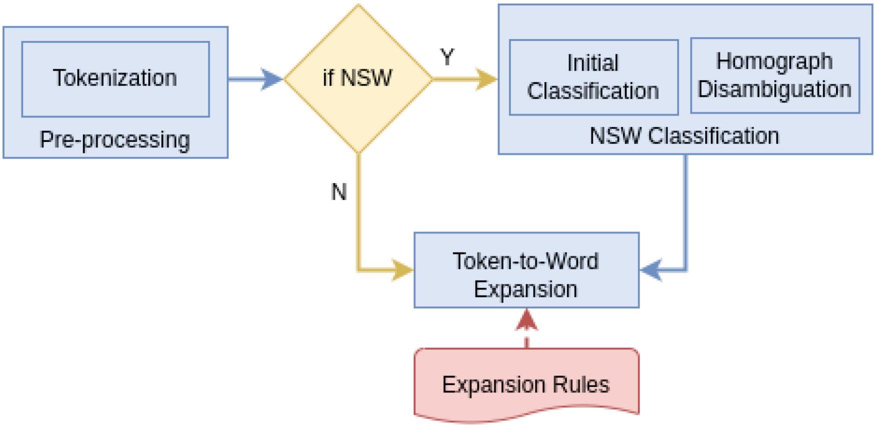


Fig. 1. Blockdiagram of text normalization system for non-standard words (NSW)

The fundamental structure of a text normalization system remains consistent across languages, with variations primarily arising from language-specific rules [6]. The proposed system tailored for Bengali TTS systems focuses on classifying non-standard words (NSWs) within the text input [1]. The text normalization process typically involves three key modules: (i) tokenization of the text into individual words, (ii) classification of the types of non-standard words, and (iii) conversion of the classified NSWs based on predefined rules and look-up tables.

2 Literature Review

There has been significant research on text normalization modules for English, Mandarin, Hindi, and other well-resourced languages [18] unlike Bengali [1, 15]. The text normalization process entails several steps, including tokenization, NSW identification, and token-to-word expansion [3,16,17]. For tokenization, most of the existing works relied on white spaces, with any leading or trailing standard punctuation being removed as features of the token [19] like they performed token-to-letter expansion based on manually crafted rules and look-up tables. Comparatively different approaches have been explored for token classification, from rule-based to trending deep-learning based approaches.

Determining the category of tokens and resolving homograph ambiguity poses significant challenges, often addressed through rule-based methodologies or data-driven approaches. The rule-based classification relies on manually crafted context-dependent rules, which, while effective, are cumbersome to develop, maintain, and adapt to new linguistic domains or languages [11,17]. Despite these challenges, most existing text normalization methods in low-resourced languages like Bengali [1] adopt rule-based strategies due to the absence of robust text analyzers and standardized databases for Indian languages. Previous endeavours to construct Bengali text normalization systems have demonstrated proficiency in classifying NSWs. However, the performance has been limited due to the limited coverage of standard databases [1]. Below, we delve into data-driven approaches for token classification and homograph disambiguation.

1. N-gram based approaches leverage bigram or trigram information tagged with parts-of-speech (POS) to address classification challenges [22]. Considering their POS tags, homograph disambiguation for NSWs is also attainable. However, these approaches need to be revised in capturing information from distant word associations, rendering them unsuitable for resolving semantic ambiguities among various NSW classes [21].
2. Conversely, Bayesian classifiers exploit long-distance word associations, irrespective of their position, to resolve semantic ambiguities. However, they may need to pay more attention to local context information and sentence structure. A comparable methodology is presented in [21], where context information is leveraged for classifying NSWs using the Winnow algorithm for homograph disambiguation.
3. Decision trees can manage intricate conditional dependencies but often struggle with large parameter spaces, particularly when it involves highly lexicalized feature sets commonly encountered in homograph resolution [22]. Yarowsky annotated the corpus with collocational distributions (word associations) occurring at different positions within sentences, collocations of lemmas (morphological roots), positional relationships extending beyond adjacency, co-occurrence within a window, and parts-of-speech tags. The discriminatory power of each piece of evidence is measured using metrics such as the

log-likelihood ratio. Panchapagesan *et al.* used decision tree-based classification for the initial classification of NSWs, while unique information regarding the NSWs was utilized for the final classification process [17].

4. Jia *et al.* employed finite state automata (FSA) and maximum entropy (ME) classifiers for classifying NSWs [11]. FSA utilizes NSW formats, employing a maximum match strategy to manage longer NSWs and effectively eliminate the necessity for tokenization. However, it requires enhanced performance for certain compound NSWs comprising elements from different classes. Accordingly, a segmentation process is applied utilizing a secondary suffix list to manage compound NSWs. In the subsequent step, maximum entropy utilizes probability estimation to minimize assumptions while adhering to imposed constraints. It employed a bound-constrained limited-memory variable metric (BLMVM) method for training parameters and the inequality smoothing algorithm within maximum entropy classifiers. Features like 4-gram sequences and heuristic indicators, including the presence of digits, whether they begin with zero, and whether they are preceded or followed by alphabetic characters, are utilized in this work.
5. In the study by Moattar *et al.* on Persian text normalization [12], various machine learning techniques, including CART (Classification And Regression Tree), SVM (Support Vector Machine), and MLP (Multi-layer Perceptron) neural networks were explored. CART, a decision tree-based method, is straightforward and interpretable. However, its accuracy heavily relies on selecting features that significantly contribute to uniquely classifying NSWs. MLP neural networks offer powerful learning capabilities but are challenging to optimize due to the extensive time required for training. Constructing the most effective MLP architecture can be complex and resource-intensive. On the contrary, SVMs typically require less training time and can achieve optimal performance more efficiently. According to Moattar *et al.*, SVMs demonstrated superior performance to CART and MLP neural networks.
6. The hybrid approach, as discussed in various studies such as [4,20,22], amalgamates the strengths of multiple techniques including Bayesian classifiers, n-grams, and decision lists. By combining these methods, the hybrid approach enhances the overall performance. One of the key benefits of the hybrid approach is its ability to capture both local and long-distance contexts. It was achieved by incorporating features and strategies from different methodologies, allowing the system to consider various levels of context and linguistic information. However, their effectiveness depends highly on the availability of standard databases annotated with discriminative features. Consequently, further research and development efforts are needed to refine these approaches and establish standardized datasets to support their implementation and evaluation.

3 Implementation

Our proposed text normalization system comprises 3 major modules: (1) preprocessing, (2) classifying NSWs, and (3) expanding tokens.

3.1 Pre-processing

Tokenization. In our text normalization system, whitespace serves as the primary delimiter between words, reflecting the conventional practice in linguistic tokenization. Additionally, punctuations such as commas (,) and semicolons (;) are utilized to delineate phrases, while full stops (.) signify the conclusion of an utterance. These delimiters are instrumental in segmenting the input text into meaningful units during tokenization. However, certain punctuations, such as the dash (-), pose unique challenges due to their variability in usage. The dash (-) is particularly interesting as it can appear within a single token. For instance, in expressions like "1-10-1974", denoting a date or "91-033-23456789", representing a phone number, the dash is an integral part of a single token. Consequently, including the dash as a tokenization delimiter may inadvertently split a single token into multiple segments, potentially introducing ambiguities in subsequent token classification tasks. Therefore, while whitespace, commas, semicolons, and full stops are employed as delimiters for tokenization purposes, special consideration is given to treating the dash (-) to ensure accurate segmentation and mitigate ambiguities in the tokenization process.

Splitting. The splitting process serves as a crucial pre-processing step in our text normalization system, aimed at enhancing the accuracy of classification and disambiguation tasks by minimizing ambiguities. This process involves breaking tokens into smaller units to a granularity level where misclassification risks are mitigated. Handling the punctuation dash (-) within tokens is central to the splitting process. Decisions regarding the dash (-) treatment are based on carefully crafted criteria to ensure optimal token segmentation. Specifically, the process considers the following decisions:

- If the token is a phone number, the dash will be deleted.
- If the token is a game score or year, whitespace will replace the dash. This means that the token is split into two or more tokens.
- If the token is dated or no information is available from the text about the class of the token, the dash will be kept for further classification.

3.2 NSW Classification

In our text normalization system, the classification module is pivotal in categorizing non-standard words (NSWs) into distinct types or classes. Initially, the module identifies NSWs and assigns them to one of six preliminary categories. Subsequently, these preliminary categories are subdivided into 14 final types or classes through further analysis and refinement, as mentioned below.

- Class 1: year/cardinal number/pin code/date
- Class 2: float/time/version
- Class 3: phone/date
- Class 4: date/fraction

- Class 5: percentage/acronym/abbreviation/day
- Class 6: email id or URLs

The classification task is accomplished using various baseline and proposed transformer-based text normalization (TN) approaches. These approaches are designed to effectively categorize non-standard words (NSWs) into appropriate classes, laying the groundwork for subsequent normalization processes. The details of these methodologies are expounded upon in the subsequent subsections, delineating the intricacies of each approach and their respective contributions to the classification task.

Rule-Based Approach. The rule-based approach is implemented through two sub-modules: initial classification and homograph disambiguation.

1. **Initial Classification**
   The preliminary classes for non-standard words (NSWs) are determined based on the following rules:
   - Class 1: year/cardinal number/pin code/date: A digit token can represent various entities like a year, cardinal number, pin code, or date. For instance, the token "1947" might denote a specific year or serve as an integer value. Similarly, in Bengali language usage, expressions like "30 baishaakha" (where "baishaakha" signifies a Bengali month) are prevalent, with the token "30" indicating a date within the context. These examples illustrate that a token comprising solely of digits is initially categorized as belonging to class 1.
   - Class 2: float/time/version: A token containing two or more digit sequences separated by a dot (.) falls into class 2. For instance, consider the token "1.22", which typically represents a fraction or version number unless contextual tokens provide time-related information. In cases where two consecutive tokens, such as "1.22 pm", are present, the first token signifies time, especially if the following token, "pm" (post meridian), indicates the second half of the day.
   - Class 3: phone/date: A token comprising solely of digit sequences separated by one or more whitespace or dashes (-) could signify a phone 0 or date. If the context includes words such as "phone", "call", "hello", "number", or "mobile" or their derivatives, the token is classified as a phone number. Alternatively, the token is categorized as a date if the contextual information pertains to a date, month, or year.
   - Class 4: date/fraction: A token with digit sequences separated by a forward slash (/) could represent either a date or a fraction. Typically, a date comprises two obliques, while a fraction utilizes only one. However, variations in distribution may occur in actual text. In the case of a date, contextual cues should provide information about the year, month, or date. Otherwise, contextual information would not pertain to year, month, or date if the token is classified as a fraction.

   - Class 5: percentage/acronym/abbreviation/day: Substitution using a look-up table is employed to manage these classes due to the requirement for increased variability between them.
   - Class 6: email ID or URLs: Identifying email IDs or URLs is straightforward as they typically include standard contextual information. For instance, a token containing alphanumeric or character sequences separated by "@" is categorized accordingly. Additionally, character sequences such as "mail", "com", or "in" further aid in their classification.

2. **Homograph Disambiguation**
   Homograph disambiguation is necessary to subdivide tokens of similar initial classes into their final classes.
   (a) Year: A class 1 token, with the context information "saal", "khr:ist:aabda", "abda" or "sana", is considered a year.
   (b) Pin code: A class 1 token with context information on "pin" and "zip" is considered a pin code.
   (c) Date: A class 1 token with context information such as a month's name after the NSW is considered a year.
   (d) Cardinal number: A class 1 token, which is not predicted as year, pin code or date, is considered a cardinal number.
   (e) Time: Class 2 tokens with contexts related to time are considered time.
   (f) Version: A class 2 token having the context information to be a version number is predicted as a version.
   (g) Float: A class 2 token, which is not considered as time or version, is predicted as a float number.
   (h) Phone: A class 3 token with context information related to a phone number is considered a phone number.
   (i) Date: A class 3 token, which is not considered a phone number with context information related to a date, is predicted as a date.
   (j) Date: A class 4 token having two obliques and context information related to date is predicted as a date.
   (k) Fraction: A class 4 token not classified as a date is a fraction.
   (l) Percentage, acronym, abbreviation and day: Due to similar structures, these NSWs are handled by substitution method using a look-up table.
   (m) Email-ids and URLs: These classes have been achieved in initial classification; therefore, further classification is avoided.

Statistical Approach. This section provides details of our proposed CRF-based statistical approach for classifying NSWs within text-to-speech synthesis systems.

- **CRF-based Approach**
  Our proposed Conditional Random Field (CRF) based classifier was motivated by its capability to effectively leverage contextual information and dependencies among neighbouring words for accurate predictions. Conditional Random Fields (CRFs) are probabilistic models used for structured prediction, which model the conditional probability of hidden (unobserved) state

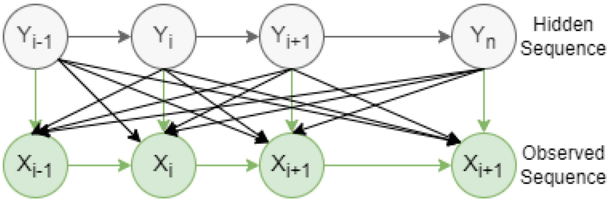


Fig. 2. Block diagram of the CRF-based TN system

sequences given observed sequences. They leverage both the observed data and the dependencies between the hidden states to make accurate predictions in sequence labeling tasks, as depicted in Fig. 2. In the present task, we considered the NSW classes, represented in sequences, while their features as the context information. This classifier was trained using an annotated corpus, where linguistic experts meticulously labelled each word with its corresponding class types. To capture the intricate details of local context, we extracted a comprehensive array of lexical, syntactic, and semantic features for each word. These features encompassed diverse aspects such as word embeddings, part-of-speech tags, and neighbouring words. With such rich contextual information, our classifier modelled a deeper understanding of the linguistic context, enhancing its ability to classify NSWs accurately. To model the sequential dependencies, we employed a linear-chain CRF architecture that is well-suited for capturing the probabilistic relationships between adjacent words in a sequence. We used a gradient-based optimization algorithm to fine-tune the model parameters to optimize the performance. This iterative optimization process ensured that the model parameters were adjusted to maximize classification accuracy and generalization performance across diverse linguistic contexts.

Deep Learning Based Approach. This section elaborates on different deep-learning approaches for text normalization in TTS systems.

- **LSTM-based Approach**
  LSTM (Long Short-Term Memory) is a sequence-to-sequence model that leverages an encoder-decoder architecture. In this methodology, words within the input text are encoded as sequences of embeddings. These word embeddings are then processed sequentially, with LSTM cells tasked with capturing both short-term and long-term dependencies and contextual information inherent within the text. Our proposed LSTM model was optimized using the backpropagation through time (BPTT) algorithm [5]. The architecture of the LSTM model is depicted in Fig. 3.
  This algorithm enabled the adjustment of LSTM parameters, ensuring that the network learns to effectively capture the nuances of the input data and

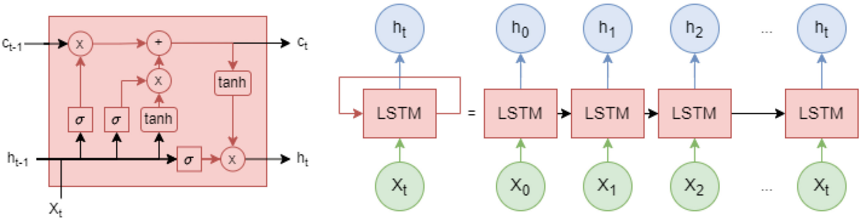


Fig. 3. Block diagram of the LSTM-based TN system

make accurate predictions. At the output layer of the LSTM model, the sigmoid activation function was applied to obtain probabilities for each class. This activation function allowed the model to output probabilities ranging from 0 to 1, facilitating the interpretation of the model's confidence in its predictions. Furthermore, we utilized binary cross-entropy loss as the objective function during training. This loss function guided the training process by measuring the discrepancy between the predicted probabilities and the actual labels. Minimizing this discrepancy, the model learned to make more accurate predictions and improve its overall performance.

- **biLSTM-based Approach**
  For this method, we implemented a time-distributed, bi-directional LSTM hidden layer. The architecture of the biLSTM model is depicted in Fig. 4. This choice of architecture allowed the LSTM to process input sequences bidirectionally, capturing both past and future contexts for each word.
  During training, we employed the backpropagation through time (BPTT) algorithm, which enabled the adjustment of LSTM parameters by propagating gradients through the entire sequence, facilitating the optimization of the

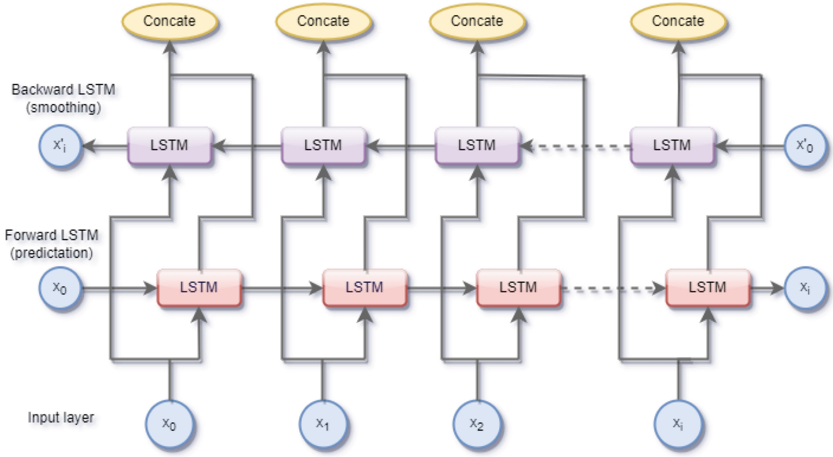


Fig. 4. Block diagram of the biLSTM-based TN system

network for accurate classification. At the output layer of the LSTM model, we applied the softmax activation function that normalized the output scores across all classes, yielding class-wise probabilities. As the objective function, we used sparse categorical cross-entropy loss that measured the discrepancy between the predicted probability distributions and the actual labels. By minimizing this loss, the model learned to make predictions that closely match the ground truth labels. Overall, the LSTM-based approach leveraged the bidirectional processing capabilities of LSTM units to capture both past and future context while utilizing BPTT and softmax activation to optimize the network for accurate classification.

- **Transformer Model**
  The Transformer Model, renowned for its attention mechanism, significantly advances traditional sequence-to-sequence models. Figure 5 illustrates the proposed transformer network architecture, embodying the sophisticated mechanisms and design principles described above. In this approach, words from the input text are tokenized and embedded into high-dimensional vectors. Positional encodings are incorporated into the embeddings to maintain sequential information, preserving the order of words within sentences. The model architecture comprises multiple encoder layers, each consisting of self-attention mechanisms and feedforward neural networks. This design enables the model to capture the input text's intricate dependencies and contextual information. During training, the model's parameters are adjusted via back-

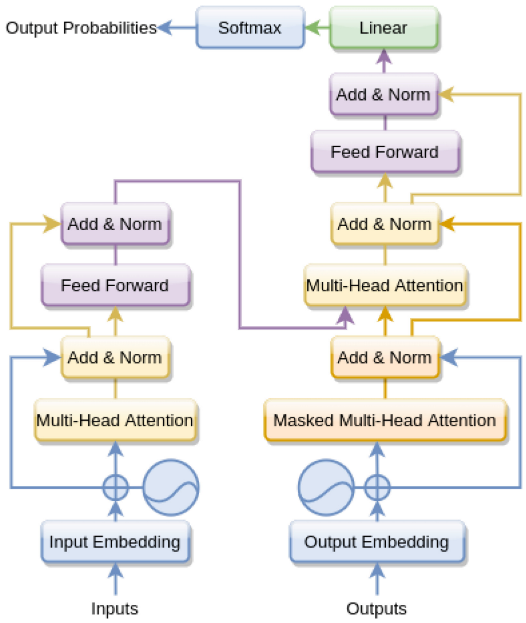


Fig. 5. Block diagram of the proposed Transformer-based TN system

propagation to minimize a suitable loss function (e.g., cross-entropy), enhancing its predictive capabilities. The self-attention mechanism allows the model to dynamically focus on different segments of the input sequence for each word, facilitating the capture of contextual nuances and dependencies within Bengali text. Additionally, position-wise feedforward networks within each transformer encoder layer enable the model to learn complex mappings from input embeddings to normalized outputs.

Our proposed model applied layer normalization after each sub-layer in the encoder, while residual connections aided information flow throughout the network, promoting training stability. The final layer of the transformer model produced probability scores for standard and non-standard labels. The softmax activation function was then applied to obtain normalized probabilities. Multi-head attention further enhanced the model's capabilities by simultaneously attending to various aspects of the input sequence.

3.3 Token-to-Word Expansion

The token-to-word expansion process generates the pronunciation for a given token based on its known class. This process relies on a combination of a look-up table and hand-crafted rules to produce accurate pronunciations.

A look-up table was utilized for straightforward tokens, where a one-to-one mapping yielded the pronunciation directly. For instance, the percentage sign "%" is replaced with its corresponding pronunciation “"শতাংশ" [/ʃɔtɐʃo/]”.

In contrast, more complex tokens, such as cardinal numbers, required the application of rules for expansion, as their pronunciation might vary. For example, while the cardinal number "1001" is pronounced as "one thousand and one", the pronunciation for "1201" becomes "one thousand two hundreds and one". In the latter case, additional hundreds necessitate a different pronunciation, illustrating the importance of rule-based expansion to represent such tokens accurately.

4 Database

We developed a corpus by extracting sentences containing at least one NSW from various sources, including news corpora, storybooks, and textbooks. A total of 4,000 sentences were gathered, encompassing 6,067 NSWs. This dataset was partitioned into training and testing subsets with 3,000 sentences with 4,585 NSWs and 1,000 sentences with 1,482 NSWs, respectively. The training data was used to develop classification rules and training. Conversely, the testing data was employed to evaluate the baseline and proposed text normalization approaches. The categorical distributions of NSWs are illustrated in Table 1.

Table 1. Corpus details: NSWs' categorical distribution

Class	Training data	Testing data	Class	Training data	Testing data
Year	561	187	Abbreviation	282	86
Cardinal number	1307	456	Acronym	201	64
Pin code	222	51	Percentage	159	53
Float	162	52	Day	308	102
Version	217	75	Date	182	56
Time	181	55	Fraction	241	61
Phone number	176	49	Mail-id or URLs	192	65

5 Evaluation

The present study evaluates the effectiveness of four baseline and one proposed text normalization (TN) approach using metrics that gauge (i) direct accuracy and (ii) improvements in text-to-speech synthesis systems.

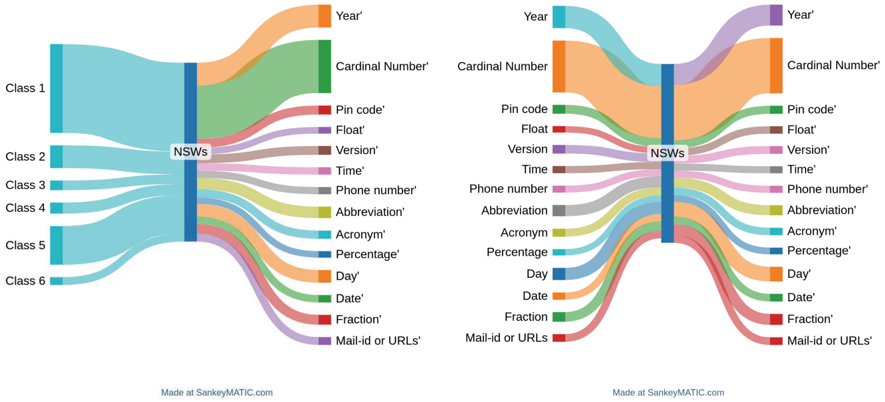
5.1 Evaluating Direct Accuracy

The performance assessment of our proposed text normalization system has been conducted using both the training and testing datasets. Table 2 provides a comprehensive overview of the proposed model's overall performance in normalizing Bengali text. We considered accuracy as the only metric following the existing approaches [18]. This evaluation encompasses various metrics and analyses, shedding light on the system's efficacy in accurately handling non-standard words (NSWs) and enhancing the quality of synthesized Bengali speech.

Table 2. Overall Accuracy of the Proposed Text Normalization Model

Model	Accuracy of Training Data	Accuracy of Testing Data
Rule-based	93.29%	89.83%
CRF-based	91.72%	83.33%
LSTM-based	94.54%	88.29%
BiLSTM-based	95.66%	90.63%
Transformer-based	**97.15%**	**91.75%**

The present rule-based approach uses a set of hand-crafted rules for classifying the NSWs. Only a few optimal rules were introduced that were specifically poised for the training data. The scope of the rules was kept general to handle Bengali NSW's structural variability. As a result, structural variability pertaining to the testing data may not be predicted. Due to that bias, the rule-based approach performed slightly better for training data. However, these gaps were absent for other approaches. Table 2 shows that NSWs with more discriminative features were accurately classified with minimum examples. As expected, the proposed transformer-based approach performed better than other baseline approaches. Such performance differences can be attributed to transformers' ability to capture lexical, syntactic, and semantic aspects out of the words from their distributions and contexts. Due to less data available for training and testing, statistical and deep-learning models could not perform as per their actual potentials. We also presented the Sankey plot for the proposed approach, showing how the initial and final NSW classes are associated in Fig. 6.



(a) Comparing overall predictions (b) Comparing initial and final predictions

Fig. 6. Analyzing proposed transformer-based TN approach

From the overall performance, transformer-based model has been the best performing approach. However, the performance could be better if could have been trained with more instances. However, the primary focus of our present study was to compare different classifiers for Bengali text normalization systems and moreover, introducing a potent domain and language-agnostic classifier that will remove the issue of extracting language-dependent features.

5.2 Evaluating Enhancements in TTS Systems

In this section, we examine the impact of the proposed text normalization (TN) approach on the performance of a Text-to-Speech (TTS) system. The baseline TTS system is an unrestricted unit-selection-based system that employs syllables as fundamental units, developed within the Festival framework [2]. To evaluate the effectiveness of our proposed TN approach, we compare it with five baseline methods and with TTS systems developed without any NSW approach. For each approach, 20 sentences are synthesized using the respective TTS systems. These synthesized sentences are then subjected to subjective evaluation for their naturalness and intelligibility. Listening tests are conducted with 20 native Bengali speakers, who rate the quality of synthesized speech on a 5-point Likert scale. Participants assign scores ranging from 1 (very bad) to 5 (very good) for each synthesized sentence, following the guidelines provided in Table 3. The collected opinion scores provide valuable insights into the perceived quality of speech synthesis across different approaches [14].

Table 3. Instruction for evaluating the quality of synthesized speech

Point	Quality of sentence
1	Poor speech with distortion and low intelligibility
2	Poor speech with distortion but intelligible
3	Good speech with less distortion and intelligible
4	Excellent speech quality with less naturalness
5	As good as natural speech

By aggregating the opinion scores provided by all participants across all sentences, a Mean Opinion Score (MOS) is computed. The impact of integrating the proposed text normalization (TN) approach into the TTS system is evaluated by assessing changes in this MOS value. We compare the mean opinion scores provided by all evaluators for each sentence before and after incorporating these approaches, as outlined in Table 4. The results indicate an overall improvement in MOS values after incorporating the proposed TN approach.

Table 4. MOS comparison of TTS systems with different TN approaches

TTS Systems	MOS
TTS system without any TN approach	3.8
TTS system with Rule-based TN approach	4.2
TTS system with CRF-based TN approach	4.1
TTS system with LSTM-based TN approach	4.1
TTS system with BiLSTM-based TN approach	4.2
TTS system with Transformer-based TN approach	**4.3**

As depicted in Table 4, integrating the proposed approach with TTS systems yielded higher Mean Opinion Scores (MOS) than other methods, aligning with our expectations. Furthermore, statistical analysis conducted using paired t-tests confirmed the significance of these improvements.

6 Conclusions and Future Work

The text normalization system in this study is crucial for text-to-speech synthesis. Tokenization is essential in the classification process, segmenting text into tokens and handling compound non-standard words (NSWs). Initial classes are strategically grouped to maximize classification accuracy. Simple optimal rules facilitate further token classification, but narrow-scoped rules are avoided due to structural variability in NSWs.

The proposed text normalization system could be improved through an exhaustive database, refined rules, and data-driven approaches. Incorporating n-gram models can enhance the system's predictive capabilities. Using statistical and deep-learning-based techniques for automatic Bengali text normalization can lead to more accurate and robust text normalization.

The transformer-based approach is domain and language agnostic, suitable for any domain and languages. It can be adapted to other languages with similar linguistic challenges, particularly those with rich morphological structures and diverse NSW types. The modular architecture allows for easy adaptation and customization to suit different language and TTS applications.

Future research should focus on expanding the dataset to include a more extensive collection of Bengali text, improving the model's contextual understanding, exploring techniques like BERT or GPT-based models, and expanding the current approach to support multilingual text normalization, including low-resource Indian languages.

References

1. Alam, F., Habib, S., Khan, M.: Text normalization system for Bangla. Technical report, BRAC University (2008)
2. Black, A.W., Lenzo, K.A.: Building synthetic voices. Language Technologies Institute, Carnegie Mellon University and Cepstral LLC **4**(2), 62 (2003)
3. Dash, N.S.: Multifunctionality of a hyphen in Bengali text corpus: problems and challenges in text normalization and POS tagging. Int. J. Innov. Stud. Sociol. Humanit. **1**(1), 19–34 (2016)
4. Garain, A., Mahata, S.K., Dutta, S.: Normalization of numeronyms using NLP techniques. In: 2020 IEEE Calcutta Conference (CALCON), pp. 7–9. IEEE (2020)
5. Ghosh, K., Halder, T., Roy, M., Biswas, C., Gayen, R.K., Chakravarty, D.: A survey on medical image diagnosis systems: problems and prospects. In: Proceedings of International Conference on Computational Intelligence, Data Science and Cloud Computing: IEM-ICDC 2021, pp. 243–252. Springer (2022)
6. Ghosh, K., Mandal, S., Roy, N.: Boosting rule-based grapheme-to-phoneme conversion with morphological segmentation and syllabification in Bengali. In: International Conference on Speech and Computer, pp. 415–429. Springer (2023)
7. Ghosh, K., Rao, K.S.: Memory-based data-driven approach for grapheme-to-phoneme conversion in Bengali text-to-speech synthesis system. In: 2011 Annual IEEE India Conference, pp. 1–4. IEEE (2011)
8. Ghosh, K., Rao, K.S.: Subword based approach for grapheme-to-phoneme conversion in Bengali text-to-speech synthesis system. In: 2012 National Conference on Communications (NCC), pp. 1–5. IEEE (2012)

9. Ghosh, K., Reddy Vempada, R., Sreenivasa Rao, K.: Phrase break prediction for Bengali text to speech synthesis system. In: ICON-2011: 9th International Conference on Natural Language Processing, Chennai, India (2011)
10. Ghosh, K., Sreenivasa Rao, K.: Data-driven phrase break prediction for Bengali text-to-speech system. In: Contemporary Computing: 5th International Conference, IC3 2012, Noida, India, 6–8 August 2012, pp. 118–129. Springer (2012)
11. Jia, Y., Huang, D., Liu, W., Dong, Y., Yu, S., Wang, H.: Text normalization in mandarin text-to-speech system. In: 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4693–4696. IEEE (2008)
12. Moattar, M.H., Homayounpour, M.M., Zabihzadeh, D.: Persian text normalization using classification tree and support vector machine. In: 2006 2nd International Conference on Information & Communication Technologies, vol. 1, pp. 1308–1311. IEEE (2006)
13. Murthy, H.A., Bellur, A., Viswanath, V., Narayanan, B., Susan, A., Kasthuri, G.: Building unit selection speech synthesis in Indian languages: an initiative by an Indian consortium. In: Proceedings of COCOSDA (2010)
14. Narendra, N., Rao, K.S., Ghosh, K., Reddy, V.R., Maity, S.: Development of Bengali screen reader using festival speech synthesizer. In: 2011 Annual IEEE India Conference, pp. 1–4. IEEE (2011)
15. Naser, A., Aich, D., Amin, M.R.: Implementation of subachan: Bengali text to speech synthesis software. In: International Conference on Electrical & Computer Engineering (ICECE 2010), pp. 574–577. IEEE (2010)
16. Olinsky, C., Black, A.W.: Non-standard word and homograph resolution for Asian language text analysis. In: INTERSPEECH, pp. 733–736 (2000)
17. Panchapagesan, K., Talukdar, P.P., Krishna, N.S., Bali, K., Ramakrishnan, A.: Hindi text normalization. In: Fifth International Conference on Knowledge Based Computer Systems (KBCS), pp. 19–22 (2004)
18. Sodimana, K., et al.: Text normalization for Bangla, Khmer, Nepali, Javanese, Sinhala, and Sundanese TTS systems. In: The 6th International Workshop on Spoken Language Technologies for Under-Resourced Languages (2018)
19. Sproat, R., Black, A.W., Chen, S., Kumar, S., Ostendorf, M., Richards, C.D.: Normalization of non-standard words. Comput. Speech Lang. **15**(3), 287–333 (2001)
20. Sproat, R., Jaitly, N.: RNN approaches to text normalization: a challenge. arXiv preprint arXiv:1611.00068 (2016)
21. Tesprasit, V., Charoenpornsawat, P., Sornlertlamvanich, V.: A context-sensitive homograph disambiguation in Thai text-to-speech synthesis. In: Companion Volume of the Proceedings of HLT-NAACL 2003-Short Papers, pp. 103–105 (2003)
22. Yarowsky, D.: Homograph disambiguation in text-to-speech synthesis. In: Progress in Speech Synthesis, pp. 157–172. Springer (1997)

Improving Sampling Methods for Fine-Tuning SentenceBERT in Text Streams

Cristiano Mesquita Garcia[1,3](✉), Alessandro Lameiras Koerich[2], Alceu de Souza Britto Jr.[3,4], and Jean Paul Barddal[3]

[1] Instituto Federal de Santa Catarina (IFSC), Câmpus Caçador, Caçador, Brazil
cristiano.garcia@ifsc.edu.br

[2] École de Technologie Supérieure (ÉTS), Université du Québec, Montréal, Canada
alessandro.koerich@etsmtl.ca

[3] Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, Brazil
{alceu,jean.barddal}@ppgia.pucpr.br

[4] Universidade Estadual de Ponta Grossa (UEPG), Ponta Grossa, Brazil

Abstract. The proliferation of textual data on the Internet presents a unique opportunity for institutions and companies to monitor public opinion about their services and products. Given the rapid generation of such data, the text stream mining setting, which handles sequentially arriving, potentially infinite text streams, is often more suitable than traditional batch learning. While pre-trained language models are commonly employed for their high-quality text vectorization capabilities in streaming contexts, they face challenges adapting to concept drift-the phenomenon where the data distribution changes over time, adversely affecting model performance. Addressing the issue of concept drift, this study explores the efficacy of seven text sampling methods designed to fine-tune language models, thereby mitigating performance degradation selectively. We precisely assess the impact of these methods on fine-tuning the SBERT model using four different loss functions. Our evaluation, focused on Macro F1-score and elapsed time, employs two text stream datasets and an incremental SVM classifier to benchmark performance. Our findings indicate that Softmax loss and Batch All Triplets loss are particularly effective for text stream classification, demonstrating that larger sample sizes correlate with improved macro F1 scores. Notably, our proposed WordPieceToken ratio sampling method significantly enhances performance with the identified loss functions, surpassing baseline results.

Keywords: Text stream · Language Model · Concept drift · Sampling methods · Fine-tuning

1 Introduction

The Internet has become part of daily life around the world. People and systems generate a vast amount of textual data through the Internet. Individuals can

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 445–459, 2025.
https://doi.org/10.1007/978-3-031-78495-8_28

chat with others, review products and services, and share comments and opinions through social media platforms, frequently working as social sensors [24]. Learning from social media posts can be relevant for institutions and governments, helping them quickly detect and respond to events [11,19], for example.

Automatically learning from textual data leveraging machine learning mechanisms brings several challenges in batch processing, such as text standardization and vectorization. Text vectorization is essential since most machine learning methods expect numeric vectors as input. Traditional vector representations, such as Bag-of-Words (BOW) [12] and Term Frequency - Inverse of Document Frequency (TF-IDF) [21], can generate very-high-dimensional vectors, which can be disadvantageous to the machine learning model, increasing the computational cost.

In a textual stream scenario, the challenges are augmented. Due to the stream characteristics, e.g., data arriving on an instance-basis or in small batches and resource limitations [3,9], generating vector representations through BOW and TF-IDF is complex. For instance, if the vector representations are generated in the first batch, new words in subsequent batches will not be represented. On the other hand, generating the representations as the batches arrive can lead to variable-dimension representations [10], a challenge since most machine learning algorithms require a fixed-dimension input.

Therefore, pre-trained language models have become popular in batch and stream scenarios due to their time-saving characteristics [10,25]. SentenceBERT (SBERT) [20] is a popular pre-trained language model specific for sentence embedding generation. Although pre-trained models save time since training a language model from scratch is costly, adjustments may be necessary for domain adaptation. In addition, changes in data distribution over time (concept drift) are frequent phenomena in real-world data and can degrade a machine-learning model's performance [9]. In textual data streams, those changes can emerge from sentiment changes, the appearance of particular words in different contexts, and so on [10]. Furthermore, computational linguistic studies using diachronic datasets attest that writing patterns change and word meanings evolve over time, e.g., semantic shift [4]. This work refers to changes as concept drift since semantic shifts are generally related to linguistic studies, which use long timespan, or diachronic, datasets and investigate those changes on a deeper, linguistic level. Therefore, to adapt to concept drift or a new domain, for example, it is important to adapt the language model. The fine-tuning process is a popular deep-learning-related process and can help adapt the language model [17,22].

Due to the computational cost of the fine-tuning process, selecting representative instances to fine-tune the language model may provide valuable information while reducing the time spent [1,23]. In this paper, we score the ability of different sampling methods in text selection for fine-tuning purposes. We also propose a sampling method, i.e., WordPieceToken ratio, whose results were promising in most scenarios evaluated. Considering the text stream setting, we assessed these methods intrinsically, i.e., in a downstream task. We also evaluated three versions of these sampling methods modified to account for the classes, totaling seven sampling methods.

The contributions of this paper are four-fold: (a) an extensive comparison among text sampling methods for fine-tuning purposes; (b) an analysis of the impact of the sampling methods considering the text stream setting; (c) an evaluation of loss functions for fine-tuning SBERT, and (d) a novel textual sampling method based on the ratio between Wordpieces and tokens of a text. The term Wordpieces represents a subword partition system present in BERT [8] that allows handling out-of-vocabulary tokens.

This paper is organized as follows: Sect. 2 presents important concepts for understanding this paper, including text streams and SentenceBERT. Section 3 presents the text-based sampling methods evaluated in this paper. Section 4 describes the experimental protocol, with datasets, settings, evaluation scenario, and results. Finally, Sect. 5 concludes this paper.

2 Background

This section presents core concepts for understanding this paper. We introduce text stream mining, SentenceBERT, and its fine-tuning process.

2.1 Text Stream Mining

According to Bifet et al. [3], data streams are "an abstraction that allows real-time analytics". In a data stream, the items arrive individually or in small sequential batches, and the stream itself can be infinite [3,9]. Different learning approaches have been developed to learn from data streams, including, for instance [9], the ability to learn incrementally, single-pass operations, elimination of input data as soon as possible after learning from it, and consumption of modest resources, i.e., processing power and time.

Text streams are a specialization of data streams in which texts arrive over time [10]. The challenges are extended in this scenario, mainly comprising natural language processing (NLP), such as text standardization, vocabulary, and representation maintenance. These NLP-related processing are challenging due to their complexity, which should meet the text stream constraints.

Frequently, text-related approaches leverage pre-trained language models [10,25]. Using pre-trained models can help save time since training a language model from scratch is computationally costly in time and resources [23]. In addition, the language model can easily be reused in different scenarios. However, an important drawback is that texts generally suffer from concept drift. Concept drift is a phenomenon frequently observed in real-world datasets and corresponds to changes in data distribution over time [9,10]. Leveraging pre-trained language models without accounting for concept drift can decrease performance in the downstream task since the texts would be represented using relatively old representations [10].

This paper leverages a pre-trained language model and evaluates the use of fine-tuning for language model updates in text stream settings, a less costly approach than training from scratch. In this paper, the selected language model is SentenceBERT [20].

2.2 SentenceBERT

SentenceBERT (SBERT) is an architecture that leverages pre-trained BERT models [20], such as BERT [8] and RoBERTa [18] models. SBERT leverages siamese networks to generate semantically meaningful representations that are compared using cosine similarity. A siamese architecture with a bi-encoder reduces the computational overhead while improving the quality of representations, compared to a cross-encoder to determine sentence similarity, as in BERT [20].

Additionally, SBERT provided significant improvements for semantic text similarity. Although the authors fine-tuned the SBERT model on natural language inference (NLI) data and also applied the model to semantic textual similarity (STS) task, SBERT demonstrated competitive results when being used as a text vectorization method for classification tasks [25].

Fine-Tuning, Data Preparation, and Loss Functions. SBERT allows several strategies for the fine-tuning process. Typically, SBERT requires texts and a label. Due to the siamese characteristic of SBERT, generally, it requires text pairs (or triplets) and a label. Depending on the strategy, this label can correspond to a class, relatedness degree between texts, or relatedness class between texts, e.g., contradiction, neural, or entailment. The loss functions and respective strategies allowed by SBERT include:

- **Batch All Triplets loss (BATL)** [14], which requires single texts and their respective classes. Internally, same-class texts are treated as positive anchors, while distinct-class texts are considered negative anchors;
- **Cosine Similarity loss (CSL)**, which expects text pairs and a cosine similarity score as a label. A clear drawback is the demand for a cosine similarity, which requires an extra method/ground truth;
- **Contrastive Tension loss (CTL)** [5], which receives single texts without labels. In this case, exact texts are treated as positive anchors, and all the texts are randomly mixed to generate negative anchors. It uses a ratio to define the number of negative anchors for each positive anchor;
- **Multiple Negative Ranking loss (MNRL)** [13] uses only positive text pairs (or triplets with a negative anchor appended) without label. In this case, texts are mixed to generate negative anchors. A noticeable characteristic is the need for a ground truth to indicate positive texts;
- **Online Contrastive loss (OCL)** requires text pairs and a label indicating their relationship. In this case, the loss function is calculated per item;

- **Softmax loss (SL)** receives text pairs and a label indicating their relationship. Reimers and Gurevych [20] leveraged this loss function for natural language inference, and therefore, the possible labels were contradiction, neutral, or entailment.

The list above is non-exhaustive. This paper aims to evaluate text-based sampling methods (see Sect. 3) to gather more useful texts for fine-tuning. Considering the above loss functions, we selected BATL, CTL, OCL, and SL. We did not select CSL and MNRL because CSL depends on a similarity measure for the text pairs and would require extra information for this calculation. MNRL, on the other hand, requires positive pairs, or triplets, with a negative anchor. Although we could leverage the classes to generate positive pairs, choosing good-quality anchors can be challenging and require deeper analysis for an assertive selection.

The high cost of training language models is well-known [23]. Although fine-tuning is cheaper than training from scratch, using all new data can also lead to high costs [1,23]. Thus, resorting to sampling methods can be beneficial in two aspects: (a) selecting more informative texts and (b) consuming fewer computational resources than using all new data.

3 Text-Based Sampling Methods

This section presents the sampling methods used in this paper for selecting texts for fine-tuning purposes. We also propose the WordpieceToken ratio and later compare it to other text-based sampling methods.

In addition to each sampling method, except for the random sampling, we evaluate an extra scenario leveraging the text labels (classes). Therefore, the sampling methods correspond to their original version and the version that accounts for the class. Algorithm 1 provides the weighted sampling pseudocode. We highlight that, optionally, the observed classes' frequencies can be used as an argument for the `WeightedSampling` function. These frequencies can also be calculated directly from the buffer, using the attribute *class*. The algorithm runs according to the following steps: (1) the buffer containing the stored items from the stream is iterated; (2) each item has its weight calculated, depending on the chosen sampling method; (3) if the classes' frequencies are considered, then the items' probabilities of less frequent classes are increased proportionally; (4) the weights are normalized; and (5) n_s instances are sampled from the buffer.

The text sampling methods employed in this paper are:

- **Length-based sampling** [1]: it ponders items by their length. This means we counted the number of tokens in each text and normalized them using the biggest and the lowest lengths. The main idea is that longer texts have more chance to encompass useful, novel information for the language model in the fine-tuning process;
- **Random sampling**: this sampling method randomly selects a given number of items from the buffer;

Algorithm 1. Algorithm of weighted sampling

```
Require: $n_s$                                   ▷ number of instances to be sampled
Require: classes_frequencies                          ▷ observed classes frequencies
Require: buffer
Ensure: buffer $\neq \emptyset$
  function WeightedSampling($n_s$, buffer, classes_frequencies: Optional)
      for each $X_i \in$ buffer do
          Calculate $X_i$.weight according to the sampling method
          if classes_frequencies $\neq$ None then
              $X_i.\texttt{weight}^* \leftarrow X_i.\texttt{weight} \times \frac{\texttt{sum(classes\_frequencies)}}{\texttt{classes\_frequencies}[X_i.\texttt{class}]}$
          end if
      end for
      Normalize weights
      Sample $n_s$ instances using the calculated $X_i.\texttt{weight}^*$ as probability
  end function
```

- **TF-IDF-based sampling**: Term Frequency - Inverse of Document Frequency (TF-IDF) [21] is a technique for measuring the importance of a given word in a document, considering a collection. The term frequency is the count of a token t in a text d. The inverse of document frequency is generally calculated as $\text{idf}(t) = \log \frac{n}{\text{document\_frequency}(t)}$, where document_frequency(t) corresponds to the number of documents containing t, and n is the total number of documents. Then, the complete TF-IDF calculation is: $\text{TFIDF}(t, d) = \text{term\_frequency}(t, d) \times \text{idf}(t)$. Given a text, the TF-IDF is calculated for each token in the text. Thus, the text's weight is the sum of the TF-IDF values of the tokens present in the text. The rationale behind this approach is to select texts with more important words across the buffer.

In addition to these sampling methods, we propose a novel text sampling method named WordpieceToken ratio.

3.1 WordpieceToken Ratio Sampling

This paper proposes a novel sampling method based on the ratio between wordpieces and tokens. Wordpiece is a technique BERT-based models use to handle out-of-vocabulary (OOV) tokens [8]. For example, the word `institutionalization` could be partitioned into two wordpieces: `[‘institutional’, ‘##ization’]`, where `##` means that there is a previous partition.

The rationale behind this sampling method is that the bigger the ratio between wordpieces and tokens, the bigger the number of unknown words (by the language model) in the text. The ratio is calculated per instance. For example, let the text instance be: “Extramedullary toxicity was limited to hypothyroidism”, a 6-token sentence. When applied to a BERT tokenizer, it is split into `[‘extra’, ‘##med’, ‘##ulla’, ‘##ry’, ‘toxicity’, ‘was’, ‘limited’, ‘to’, ‘h’, ‘##yp’, ‘##oth’, ‘##yr’, ‘##oid’, ‘##ism’]`, resulting in 14 wordpieces. Thus, this instance's weight is $\frac{14}{6} = 2.33$. After all the text instances from the buffer have their weights calculated, several instances are sampled considering the instances' weights and class frequencies according to Algorithm 1.

Therefore, we hypothesize that sampling texts by weighting the Wordpiece-Token ratio may retrieve texts with more useful information for the fine-tuning process.

4 Experimental Results

This section provides the experimental results, including the experimental protocol, datasets, proposed scenario, evaluation metrics, and the results.

4.1 Experimental Protocol

Datasets. Two datasets were used: Airbnb and Yelp.

- **Airbnb:** This dataset was obtained from the Inside Airbnb[1], considering data related to the New York City. Since New York City is one of the most popular destinations in the United States[2], the Airbnb dataset related to New York City has several reviews, enough for multiple sampling for text stream simulation (see Sect. 4.2). This dataset, by default, is not ready for classification tasks. Therefore, we leveraged: (a) a pre-trained model for language identification is used (lid.176.ftz [15,16]), and (b) a pre-trained model for sentiment analysis to infer the reviews' sentiment (Twitter RoBERTa Base Sentiment[3] model [2]). Thus, the English reviews were filtered, and their sentiments were inferred. The sentiments, i.e., positive, negative, and neutral, were used as labels in the classification task. The processing steps are available on Github[4].
- **Yelp:** This dataset is provided on Yelp Datasets[5]. Yelp consists of reviews collected regarding over 130 thousand businesses. These reviews are accompanied by a category from a scale of stars between 1 and 5. This category is used as a label in the text classification task.

An essential characteristic regarding the above datasets is the presence of a timestamp, which is crucial for text stream simulation. The data distributions are presented in Fig. 1. Noticeably, the datasets are imbalanced, reflecting the nature of real-world data.

Other relevant information on the experiments are:

- **Sample sizes:** we considered four sample sizes: 500, 1000, 2500, and 5000. Those values were chosen since they represent between 1% and 10% of the buffer size, which can be considered reasonable values;

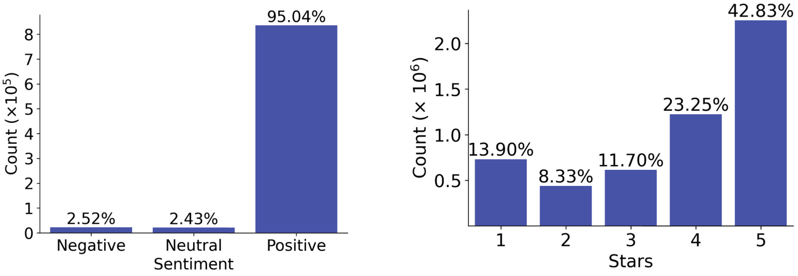
[1] http://insideairbnb.com/get-the-data.

[2] Available at: https://www.cntraveler.com/story/most-visited-american-cities. Accessed on Jan 20th, 2024.

[3] Available at: https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment.

[4] https://github.com/cristianomg10/methods-for-generating-drift-in-text-streams.

[5] https://www.yelp.com/dataset.



(a) Class distribution of Airbnb dataset after filtering.

(b) Class distribution of Yelp dataset. Adapted from [25].

Fig. 1. Class distribution of the datasets used in this paper

- **Classifier:** we selected the Incremental Support Vector Machine (ISVM) as the classifier because it is updated incrementally according to the arrival of new data. As its batch counterpart [6], ISVM calculates an optimal hyperplane between instances of two distinct classes, and the surrounding instances are assumed as support vectors [7].
- **Hardware:** the hardware used in the experiments is a 13th Gen Intel(R) Core(TM) i9-13900K, 128 GB of RAM running Ubuntu 22.04 LTS, and 2 x GPU GeForce RTX 4090 (24 GB).

4.2 Proposed Scenario

Considering the text stream mining setting, the proposed scenario is run as follows: (1) a text stream of length 200,000 sampled (stratified by class/label) from the original datasets; (2) the text stream classification is performed one-by-one; (3) a buffer accumulates the first 50,000 items of the text stream; (4) at the moment $t = 50,000$, a sampling method, among the methods presented in this paper, will sample a predefined number of items from those described in Sect. 4.1; (5) after sampling, the fine-tuning process for the language model, i.e., the SBERT model, is triggered using the sampled texts and the selected loss function; and (6) the evaluation metrics are calculated cumulatively in a test-then-train fashion, i.e., in a prequential manner.

This process was executed five times per sampling method and loss function. In each run, the entire stream was also sampled from the original dataset in a stratified manner. We used $t = 50,000$ as the effects of the fine-tuning and sampling methods would be easier to spot than at the end of the stream.

4.3 Loss Functions Settings

Considering the loss functions presented in Sect. 2.2, their inputs for fine-tuning were defined as follows:

- **BATL:** single texts and their respective classes;
- **CTL:** single texts;
- **OCL:** text pairs and a label, which we adapted for considering the distance between classes by calculating: label = $1 - \frac{abs(X_1.\text{class} - X_2.\text{class})}{|classes|}$, where X_ϕ is a text, $| \cdot |$ is the cardinality, and abs($\cdot$) is the absolute value. Thus, label is one if the classes are the same, indicating the similarity;
- **SL**: similarly to OCL, SL receives text pairs and a label, which we adapted to be the absolute distance, calculated as label = abs(X_1.class − X_2.class), where X_ϕ is a text, and abs($\cdot$) is the absolute value.

This paper leverages the pre-trained SBERT model `paraphrase-MiniLM-L6--v2`. When fine-tuning, we used 32-sized batches, 10 epochs, and 100 warmup steps, which are values frequent in the documentation.

4.4 Evaluation Metrics

Given the class imbalance of the datasets used in experimentation, results regarding Macro F1 Score and elapsed time were reported. In particular, the Macro F1 Score averages the harmonic mean of precision and recall obtained per class.

4.5 Results

Considering the presented scenario, loss functions, and sampling methods, the results obtained regarding Macro F1-Scores are demonstrated in Fig. 2. For readability issues, we kept different scales on the y-axis. The dashed lines correspond to the maximum and minimum Macro F1-Score values using SBERT without update and were used as a baseline. The x-axis regards the sampling methods, while the boxes correspond to the tested sample sizes. The Macro F1-Scores obtained for the baseline were: (a) 75.13 ± 0.28 (min: 74.86; max: 75.50) for the Airbnb dataset, and (b) 44.60 ± 0.12 (min: 44.46; max: 44.78) for Yelp dataset. Regarding the elapsed times, the values obtained were (in seconds): (a) 496.78 ± 0.70 (min: 495.89; max: 497.86) for the Airbnb dataset, and (b) 558.82 ± 3.19 (min: 553.36; max: 561.10) for Yelp dataset.

Considering the results obtained for the Airbnb dataset, we can see that the loss function is crucial for improving the results. CTL and OCL performed worse than when there were no updates, according to the dashed lines. Considering OCL and CTL, only the proposed WordPieceToken ratio sampling method (using class in the sampling, sample size 500, and CTL) performed equivalently to SBERT without update. Apart from this point, all the combinations for CTL and OCL performed worse than SBERT without update. Although sometimes equivalent to the competitors, the proposed WordPieceToken ratio (class) generally obtains higher averages than its peers with the same sample size, except for using the OCL. Furthermore, increasing the sample size for these loss functions degraded the results. We assume that these loss functions, together with the looseness of anchoring, ease the model to suffer from catastrophic forgetting.

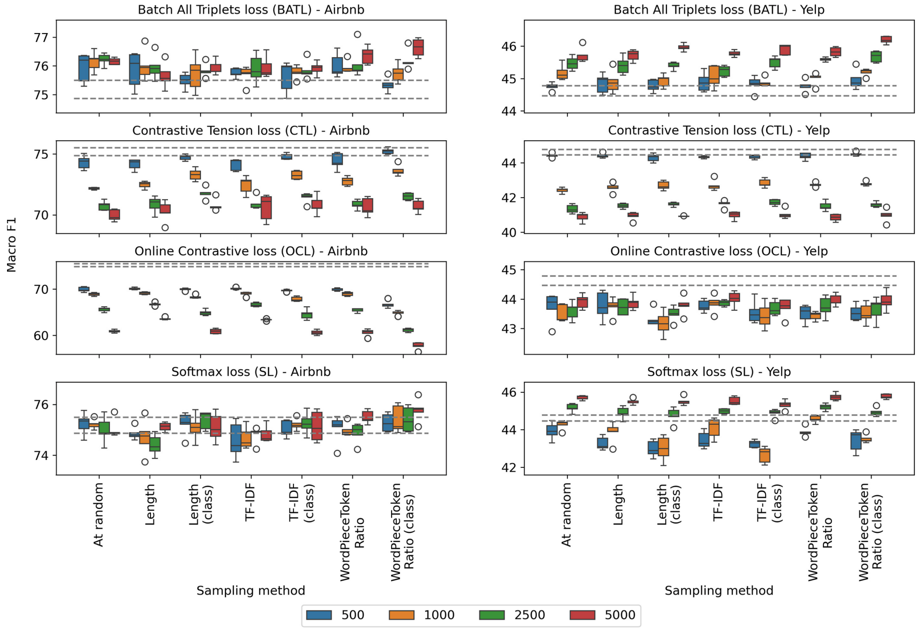


Fig. 2. Results for Airbnb (left) and Yelp (right). The dashed lines correspond to the maximum and minimum Macro F1-Score values using SBERT without update. The numbers in the legend correspond to the sample sizes.

Still analyzing the results for the Airbnb dataset, regarding BATL and SL, mostly in SL, all sampling methods were equivalent to SBERT without updates. However, using the Length or TF-IDF sampling method without accounting for the classes led to smaller Macro F1-Scores. In addition, it is possible to notice that, in the SL case, accounting for the classes in the sampling method helped reach a better Macro F1-Score than the same sampling method without accounting for the class. In the BATL scenario, all methods perform similarly across the sample sizes, except for the WordPieceToken ratio, which obtained the highest Macro F1-Score using a sample size of 5000. Therefore, using a sample size of 5000 improved the performance in these cases.

Similar observations can be made by switching to the results concerning the Yelp dataset: CTL and OCL led to poorer results than BATL and SL. CTL provided the worst results for this dataset; the bigger the sample size, the worse the performance. Again, CTL seems to lead to catastrophic forgetting. It can be credited to its simple way of generating text pairs for fine-tuning. On the other hand, OCL obtained equivalent results across sampling methods and sample sizes, but all were worse than SBERT without updates. In addition, considering the sample sizes of 2500 and 5000, all methods showed increased performance.

BATL and SL functions led to increased performance for the Yelp dataset compared to the baseline (dashed lines). For BATL, all sampling methods using

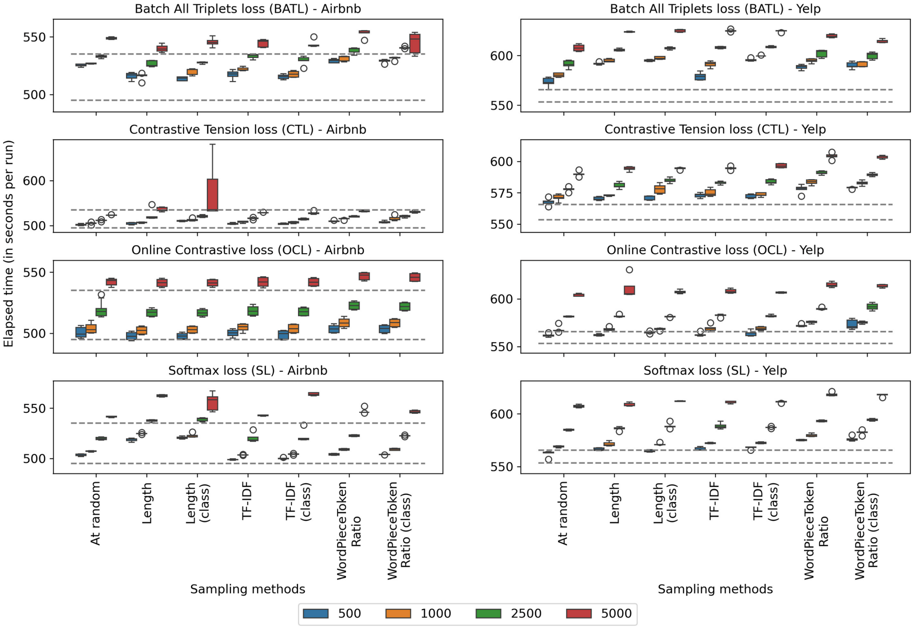


Fig. 3. Elapsed times for Airbnb (left) and Yelp (right). The dashed lines correspond to the maximum and minimum Macro F1-Score values using SBERT without update.

the sample size of 500 and 1000 are equivalent to the baseline. From the sample size of 1000, the At Random and WordPieceToken ratio sample methods reached Macro F1-Score values superior to the baseline. Regarding SL, most sampling methods are superior to the baseline from the sample size of 2500. Smaller samples led to decreased performance compared to the baseline.

Regarding the elapsed times, Fig. 3 shows measured values in each setting. Again, dashed lines correspond to the baseline, i.e., SBERT without update. For the Airbnb dataset using the BATL function, we noticed that the proposed WordPieceToken ratio took longer than the baseline to run, essentially from the sample size of 2500. For CTL, all methods, except for the length sampling method in both variations, took longer than the baseline. Specifically for Length (class), it was unstable, taking a reasonable time when using the sample size of 5000. It somehow can be expected since longer reviews would be selected, and the fine-tuning process would take longer. However, this should also be visualized for the Length sampling method, but it has not happened. We hypothesize that an anomaly in the GPU may have led to this increased variation. In OCL, all sampling methods have similar elapsed times: only sampling 5000 items led to higher run times than the baseline. At last, for SL, the Length- and TF-IDF-based methods took longer than the baseline from the sample size of 2500. In addition, Length (class) showed similar behavior to CTL regarding instability.

Table 1. Condensed results. Bold values are the best values obtained per row. Values in yellow (Macro F1) and green (elapsed time) are the best per dataset/sample size.

		At random		Length		Length (class)		TF-IDF		TF-IDF (class)		WordPieceToken ratio		WordPieceT. (class)	
Dataset	**LF**	**Macro F1**	**Elaps. time**	**Macro F1**	**Elaps. time**	**Macro F1**	**Elaps. time**	**Macro F1**	**Elaps. time**	**Macro F1**	**Elaps. time**	**Macro F1**	**Elaps. time**	**Macro F1**	**Elaps. time**
Airbnb	BATL	75.94±0.51	525.29±1.09	75.86±0.63	516.03±3.20	75.49±0.27	**513.74±1.74**	75.75±0.16	517.22±3.97	75.63±0.55	515.42±2.17	**76.01±0.43**	529.26±1.71	75.34±0.25	529.00±1.55
(500)	CTL	74.31±0.57	**502.06±1.84**	74.13±0.45	504.31±2.31	74.72±0.24	510.63±1.34	74.12±0.48	504.55±1.81	74.76±0.25	503.81±1.52	74.39±0.68	510.04±1.32	**75.24±0.24**	508.18±2.29
	OCL	69.92±0.49	500.51±4.21	**70.13±0.22**	497.78±3.27	69.93±0.26	**497.78±2.52**	70.12±0.21	500.30±2.81	69.70±0.21	498.81±3.43	69.95±0.28	503.60±3.05	66.71±0.74	503.60±3.23
	SL	75.24±0.44	503.34±1.12	74.86±0.25	518.56±1.62	75.25±0.47	520.60±1.19	74.58±0.72	**498.92±0.48**	75.03±0.33	499.92±0.75	75.06±0.57	504.14±0.86	**75.29±0.36**	503.97±0.27
Airbnb	BATL	**76.15±0.35**	526.95±0.48	76.00±0.53	**515.73±3.28**	75.74±0.62	519.87±2.88	75.70±0.32	522.05±1.64	75.79±0.22	517.72±2.71	75.95±0.21	531.35±2.33	75.75±0.33	531.44±1.53
(1000)	CTL	72.16±0.09	**505.08±2.57**	72.46±0.29	507.00±1.24	73.30±0.49	512.99±2.82	72.44±0.73	508.19±2.09	73.25±0.35	507.29±1.95	72.76±0.37	515.10±1.81	**73.64±0.45**	516.06±5.47
	OCL	68.91±0.32	503.90±3.84	**69.12±0.25**	**502.31±3.06**	68.31±0.37	502.92±2.82	68.97±0.43	504.76±3.00	67.86±0.49	503.85±3.45	68.96±0.39	508.66±3.74	64.88±0.46	508.48±3.26
	SL	75.23±0.20	507.20±0.53	74.71±0.70	524.81±0.69	75.09±0.51	522.96±2.10	74.67±0.44	**503.55±0.41**	75.21±0.23	504.27±0.56	75.00±0.27	508.90±0.87	**75.42±0.55**	509.07±1.04
Airbnb	BATL	**76.23±0.21**	532.92±1.24	75.96±0.44	**526.52±2.68**	75.84±0.24	527.48±0.85	75.90±0.51	533.20±2.16	75.85±0.33	529.35±4.03	76.10±0.57	538.23±2.76	76.20±0.35	540.58±0.78
(2500)	CTL	70.76±0.39	**511.41±1.76**	70.84±0.68	523.80±12.93	**71.78±0.47**	520.68±2.58	70.96±0.52	516.08±1.99	71.41±0.43	514.46±1.72	70.84±0.38	520.30±1.42	71.51±0.29	520.60±2.34
	OCL	65.61±0.47	519.29±6.45	**66.71±0.31**	516.95±2.99	64.91±0.59	**516.74±2.84**	66.63±0.41	518.33±3.89	64.56±1.05	517.66±3.32	65.46±0.42	522.70±3.29	61.16±0.37	521.83±3.31
	SL	75.00±0.53	**519.83±1.29**	74.40±0.42	537.47±1.17	75.39±0.36	538.80±1.83	74.82±0.33	520.59±4.74	75.25±0.44	521.87±6.34	74.89±0.40	522.76±0.90	**75.41±0.50**	522.59±0.59
Airbnb	BATL	76.16±0.12	548.77±0.93	75.66±0.44	**539.97±3.28**	75.94±0.26	545.38±3.80	75.95±0.38	544.53±3.41	75.91±0.23	543.80±3.49	76.37±0.31	552.95±3.40	**76.64±0.32**	544.69±9.47
(5000)	CTL	69.94±0.47	**523.54±0.68**	70.27±0.86	536.06±4.73	70.77±0.49	577.01±65.48	70.62±1.10	528.92±0.60	**70.81±0.78**	528.31±3.22	70.63±0.74	532.11±1.80	70.70±0.53	530.71±2.38
	OCL	60.90±0.29	541.87±2.84	63.61±0.24	541.25±3.14	60.90±0.52	**541.18±2.90**	**63.37±0.18**	541.97±3.90	60.62±0.50	541.88±3.27	60.61±0.73	546.76±2.96	57.73±0.75	545.78±3.48
	SL	75.02±0.38	**541.39±0.54**	75.12±0.17	562.32±1.08	75.10±0.57	556.26±8.92	74.83±0.34	542.58±0.51	75.13±0.61	563.87±1.56	75.50±0.27	546.98±2.86	**75.77±0.45**	546.37±1.39
Yelp	BATL	44.75±0.12	**573.14±5.07**	44.85±0.31	591.92±1.27	44.78±0.19	594.89±0.87	44.89±0.30	579.11±3.89	44.83±0.25	595.63±1.35	44.77±0.19	588.47±2.56	**44.96±0.31**	590.06±3.50
(500)	CTL	44.43±0.11	**567.70±2.88**	44.42±0.14	570.24±1.42	44.30±0.23	571.04±1.79	44.33±0.07	572.89±1.88	44.32±0.11	571.94±1.63	44.37±0.19	577.88±3.47	**44.51±0.11**	578.93±0.70
	OCL	43.73±0.50	**561.79±1.79**	**43.77±0.49**	562.14±1.05	43.33±0.28	564.83±1.13	43.76±0.21	562.65±2.03	43.55±0.39	564.05±2.98	43.50±0.31	572.15±1.28	43.54±0.27	573.24±5.18
	SL	**43.93±0.46**	**562.21±2.95**	43.27±0.35	566.43±1.35	42.98±0.44	564.91±1.02	43.45±0.46	567.24±1.46	43.25±0.20	567.83±1.32	43.89±0.25	575.07±0.55	43.43±0.60	576.29±2.29
Yelp	BATL	45.18±0.25	**580.35±2.03**	44.89±0.36	594.96±1.61	44.93±0.20	597.51±1.41	45.05±0.34	591.17±3.14	44.87±0.14	599.75±0.48	45.00±0.18	594.88±2.09	**45.19±0.11**	590.76±2.66
(1000)	CTL	42.42±0.16	**571.09±2.78**	42.56±0.26	572.81±0.45	42.69±0.25	578.06±3.92	42.70±0.31	575.08±3.13	**42.83±0.25**	573.45±1.60	42.74±0.11	583.48±2.05	42.79±0.12	582.68±1.84
	OCL	43.50±0.30	**568.04±3.69**	43.74±0.31	568.48±1.65	43.17±0.42	568.33±1.18	**43.85±0.29**	569.32±3.51	43.43±0.44	569.00±1.84	43.43±0.15	575.65±1.27	43.52±0.34	575.20±1.30
	SL	44.27±0.26	**568.90±0.80**	43.86±0.54	571.41±2.11	43.00±0.67	571.23±1.02	44.10±0.55	572.35±0.61	42.66±0.44	572.53±0.84	**44.57±0.20**	579.77±1.38	43.51±0.23	582.36±2.09
Yelp	BATL	45.47±0.21	**591.20±3.90**	45.40±0.28	605.69±1.19	45.39±0.11	607.26±1.19	45.23±0.17	608.03±1.13	45.50±0.19	608.62±0.88	45.59±0.06	602.05±3.93	**45.68±0.17**	599.80±3.57
(2500)	CTL	41.34±0.25	**577.66±1.80**	41.52±0.14	580.76±2.51	41.61±0.11	584.84±2.03	41.64±0.21	582.74±1.18	**41.71±0.17**	583.70±1.95	41.50±0.27	590.90±1.40	41.60±0.15	589.44±1.23
	OCL	43.60±0.31	581.54±0.52	43.73±0.27	581.98±1.18	43.50±0.26	**581.37±0.40**	**43.86±0.11**	582.63±1.47	43.68±0.26	582.26±1.07	43.74±0.35	589.63±1.17	43.65±0.41	591.88±3.68
	SL	45.16±0.23	**584.80±0.75**	44.90±0.33	585.97±1.66	44.75±0.42	588.60±2.74	44.98±0.14	588.49±2.93	44.88±0.23	587.16±0.77	**45.20±0.18**	593.36±0.75	44.93±0.22	594.54±1.03
Yelp	BATL	45.74±0.23	**607.66±3.46**	45.71±0.18	623.79±0.50	45.97±0.11	625.05±1.34	45.77±0.09	624.96±1.02	45.90±0.16	624.53±0.88	45.82±0.15	619.65±1.68	**46.19±0.12**	614.65±1.70
(5000)	CTL	40.88±0.27	**590.04±2.02**	40.93±0.23	594.08±1.76	40.92±0.02	594.33±0.79	40.98±0.23	594.59±1.29	**41.04±0.27**	596.56±1.70	40.84±0.20	604.23±2.38	40.98±0.37	603.21±1.27
	OCL	43.91±0.24	**604.26±1.40**	43.87±0.24	611.76±11.11	43.79±0.32	607.45±1.89	**44.00±0.26**	608.11±2.32	43.71±0.32	606.64±0.69	43.96±0.20	614.86±2.68	43.94±0.32	613.43±1.70
	SL	45.71±0.12	**607.34±1.31**	45.50±0.15	608.84±1.61	45.51±0.22	612.08±0.34	45.53±0.22	610.97±1.14	45.32±0.25	611.40±0.81	45.75±0.20	618.60±1.74	**45.76±0.13**	617.86±1.26

Regarding the Yelp dataset, considering the BATL function, random sampling led to shorter elapsed times. Length- and TF-IDF-based (with class) sampling methods led to longer elapsed times. Differently, in CTL, the proposed WordPieceToken ratio sampling reached the highest run times compared to all other methods. The same behavior happens in the SL function. For the OCL, the elapsed times for all methods are equivalent to the ones from SBERT without update, except for the WordPieceToken ratio sampling method.

Table 1 condenses the results obtained. The bold values are the best per dataset, sample size, and loss function (LF) combination, i.e., per row. The values in yellow and green are the best Macro F1 scores and elapsed times per dataset/sample size. WordPieceToken ratio (class) obtained the best Macro F1-Scores in 5 out of 8 dataset/sample size pairs, and WordPieceToken ratio in 1 out of 8. Sampling at random was the fastest in 6 out of 8 dataset/sample size pairs. Furthermore, the best Macro F1-Scores increase with the sample size.

Although elapsed time is important and variations followed similar patterns, i.e., the higher the sample size, the longer the elapsed time, the differences between them show that they may not impede fine-tuning in text stream scenarios, depending on the hardware. The differences between maximum and minimum elapsed times are 187.07 s for Airbnb and 73.54 s for Yelp, which represent 37% and 13% of their respective average elapsed times.

5 Conclusion

Learning from text streams is challenging due to the constraints of text streams, such as time, resources, and one-pass processing [9,10]. Furthermore, the existence of concept drifts in texts produced over time is well-known. A way to overcome textual concept drifts is by updating the language model. Updating (or fine-tuning) the language model is generally costly if all new data is considered.

This paper evaluates four different sampling methods, i.e., random sampling, length sampling, TF-IDF sampling, and WordPieceToken ratio sampling (proposed in this paper), where the latter three had a version that accounted for instances' classes, aiming at sampling important, informative texts from the buffer. Four loss functions were assessed in combination with the text sampling methods, i.e., Batch All Triplets loss (BATL), Contrastive Tension loss (CTL), Online Contrastive loss (OCL), and Softmax loss (SL).

We observed that the loss function plays a crucial role in improving the performance of the text classification task. Considering the scenarios assessed, CTL and OCL functions were insufficient to achieve satisfactory performance levels. On the other hand, BATL and SL functions aided in maintaining interesting performance levels, sometimes above the SBERT without update (our baseline), suggesting the SBERT can benefit from these loss functions. In addition, the elapsed times were comparable to the baseline. Therefore, BATL and SL were the most suitable functions for text stream classification among those evaluated in this paper. However, one must be careful when employing this approach because fine-tuning can create a bottleneck in real-time applications. Finally, text sampling methods are also crucial in fine-tuning. Our experiments suggest that the proposed WordPieceToken ratio method, especially leveraging the instances' classes, can retrieve more informative texts and favor the machine learning model's performance after fine-tuning.

In future works, we intend to (a) extend to different stream mining tasks, (b) identify proper moments to automatically trigger the fine-tuning, (c) evaluate the balance between the sampling method and the resources consumption, and (d) evaluate other pre-trained BERT-based models with the combinations of loss functions and sampling method used in this paper.

Acknowledgements. Cristiano Mesquita Garcia is a grantee of a *Doutorado Sanduíche* scholarship provided by *Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES). This paper is also a byproduct of a CNPq project coordinated by Prof. Jean Paul Barddal under grant number 409371/2022-0 and is tied to CNPq International #441610/2023-4.

References

1. Amba Hombaiah, S., et al.: Dynamic language models for continuously evolving content. In: Proceedings of the 27th ACM SIGKDD, pp. 2514–2524 (2021)
2. Barbieri, F., et al.: TweetEval: unified benchmark and comparative evaluation for Tweet classification. In: Findings of the ACL: EMNLP (2020)
3. Bifet, A., et al.: Machine Learning for Data Streams: With Practical Examples in MOA. MIT Press, Cambridge (2023)
4. Bravo-Marquez, F., et al.: Incremental word-vectors for time-evolving sentiment lexicon induction. Cogn. Comput. 1–17 (2022)
5. Carlsson, F., et al.: Semantic re-tuning with contrastive tension. In: Int. Conf. on Learning Repr. (2020)
6. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**, 273–297 (1995)
7. D'Andrea, E., et al.: Monitoring the public opinion about the vaccination topic from tweets analysis. Expert Syst. Appl. **116**, 209–226 (2019)
8. Devlin, J., et al.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference on of the North American Chapter of the ACL: Human Language Technology (2019)
9. Gama, J., et al.: A survey on concept drift adaptation. ACM CSUR. **46**(4), 1–37 (2014)
10. Garcia, C.M., Abilio, R.S., Koerich, A.L., Britto Jr, A.d.S., Barddal, J.P.: Concept drift adaptation in text stream mining settings: a systematic review. ACM Trans. Intell. Syst. Technol. (2024). https://doi.org/10.1145/3704922
11. Garcia, C.M., et al.: Event-driven sentiment drift analysis in text streams: an application in a soccer match. In: Proceedings of the 22nd International Conference on Machine Learning and Applications (ICMLA) (2023)
12. Harris, Z.S.: Distributional structure. Word **10**(2–3), 146–162 (1954)
13. Henderson, M., et al.: Efficient natural language response suggestion for smart reply. arXiv preprint arXiv:1705.00652 (2017)
14. Hermans, A., et al.: In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737 (2017)

15. Joulin, A., et al.: Bag of tricks for efficient text classification. In: Proceedings of the 15th Conference of the Europe Chapter of the ACL, vol. 2, Short Papers (2016)
16. Joulin, A., et al.: FastText.zip: compressing text classification models. arXiv:1612.03651 (2016)
17. Lee, J., et al.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics **36**(4), 1234–1240 (2020)
18. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
19. Pohl, D., et al.: Batch-based active learning: application to social media data for crisis management. Exp. Syst. with Appl. (2018)
20. Reimers, N., Gurevych, I.: Sentence-BERT: sentence embeddings using siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Method in NLP. ACL (2019)
21. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Inf. Process. Manage. **24**(5), 513–523 (1988)
22. Schneider, E.T.R., et al.: CardioBERTpt: transformer-based models for cardiology language representation in portuguese. In: 2023 IEEE 36th International Symposium on Computer-Based Medical System (CBMS), pp. 378–381. IEEE (2023)
23. Sharir, O., et al.: The cost of training NLP models: a concise overview. arXiv preprint arXiv:2004.08900 (2020)
24. Suprem, A., Pu, C.: ASSED: a framework for identifying physical events through adaptive social sensor data filtering. In: Proceedings of the 13th ACM International Conference on Distributed and Event-based System, pp. 115–126 (2019)
25. Thuma, B.S., et al.: Benchmarking feature extraction techniques for textual data stream classification. In: 2023 International Joint Conference on Neural Network (IJCNN), pp. 1–8. IEEE (2023)

Enhancing Authorship Attribution Through Embedding Fusion: A Novel Approach with Masked and Encoder-Decoder Language Models

Arjun Ramesh Kaushik(✉), R. P. Sunil Rufus, and Nalini Ratha

University at Buffalo, The State University of New York, Getzville, USA
{kaushik3,sunilruf,nratha}@buffalo.edu

Abstract. The increasing prevalence of AI-generated content alongside human-written text underscores the need for reliable discrimination methods. To address this challenge, we propose a novel framework with textual embeddings from Pre-trained Language Models (PLMs) to distinguish AI-generated and human-authored text. Our approach utilizes Embedding Fusion to integrate semantic information from multiple Language Models, harnessing their complementary strengths to enhance performance. Through extensive evaluation across publicly available diverse datasets, our proposed approach demonstrates strong performance, achieving classification accuracy greater than 96% and a Matthews Correlation Coefficient (MCC) greater than 0.93. This evaluation is conducted on a balanced dataset of texts generated from five well-known Large Language Models (LLMs), highlighting the effectiveness and robustness of our novel methodology.

Keywords: Authorship Attribution · Large Language Models · Generative AI

1 Introduction

In recent years, the landscape of natural language generation (NLG) technology has undergone remarkable advancements, revolutionizing the diversity, control, and quality of texts generated by Large Language Models (LLMs). Notably, OpenAI's ChatGPT, Google's Gemini, and Meta's Llama stand out as prime examples, showcasing exceptional performance across a myriad of tasks, including answering questions, composing emails, essays, and even code snippets. However, while these advancements herald a new era of human-like text generation at unprecedented efficiency, they also bring to the forefront pressing concerns regarding the detection and mitigation of potential misuse of LLMs. While there are many issues with LLMs in terms of their hallucinating responses and toxic language, the newfound capability of LLMs to emulate human-like text raises significant apprehensions about their potential misuse in activities in many areas such as phishing, disinformation campaigns, and academic dishonesty. Instances

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 460–471, 2025.
https://doi.org/10.1007/978-3-031-78495-8_29

abound where educational institutions have resorted to banning ChatGPT due to apprehensions regarding its potential for facilitating cheating in assignments [2,20], while media outlets have sounded the alarm over the proliferation of fake news generated by LLMs [18]. Such concerns surrounding the misuse of LLMs have cast a shadow over their application in critical domains such as media and education.

Accurate detection of LLM-generated texts emerges as a pivotal requirement for realizing the full potential of NLG technology while mitigating the potentially serious consequences associated with its misuse. From the perspective of end-users, the ability to discern between human-authored and LLM-generated text holds the promise of bolstering trust in NLG systems and fostering wider adoption. For developers and researchers in the realm of Machine Learning, effective text detection mechanisms can aid in tracking generated texts and thwarting unauthorized usage.

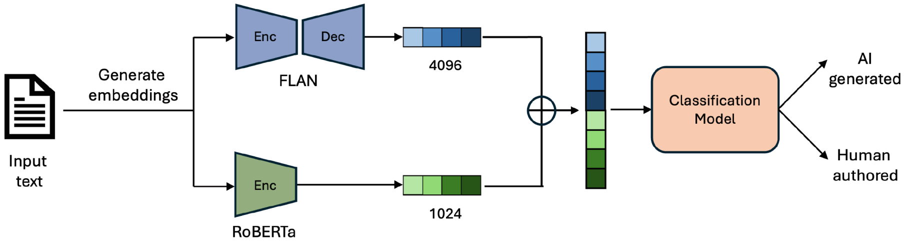


Fig. 1. An overview of our framework to perform authorship attribution between machine-generated and human-authored texts.

Given the critical significance of accurate LLM-generated text detection, we propose a novel framework based on representing text as images through Embedding Fusion as in Fig. 1. Embeddings are low-dimensional representations of high-dimensional inputs like text, aiming to capture similarities by positioning related inputs closely in the embedding space, thereby enabling AI systems to comprehend inputs akin to human understanding. Inspired by [7], we combine textual representations across Pre-trained Language Models (PLMs) and leverage their complementary strengths. Additionally, post-concatenation of the feature vectors, we reshape the fused feature vector into a 2D representation to capture the inter-embedding relationships better. Amongst the 3 types of PLMs - Autoregressive Language Models (ALMs), Masked Language Models (MLMs), and Encoder-Decoder Language Models (EDLMs) [12], our experiments indicate that the combination of the former 2 PLMs works best. We believe this is because ALMs are trained with a focus on content generation and more emphasis is placed on the final tokens. With thorough experimentation using publicly available datasets, we validate that our framework achieves better results than the state-of-the-art (SOTA) methods for the task of discerning AI-generated and human-authored text.

The rest of the paper is structured as follows. We dive into the past and recent research contributions in solving the authorship attribution task in Sect. 2. A detailed overview of the dataset is shown in Sect. 3. Sections 4 and 5 provide the methodology and results in support of our approach.

2 Related Work

The concept of authorship attribution using BERT embeddings has been effectively demonstrated by many researchers. PART [5] and BertAA [3] shows how BERT embeddings can be used to grasp authors' writing styles and generate stylometric representations. [15] explores Siamese Networks for authorship attribution (AA), comparing their effectiveness with BERT fine-tuning. On the other hand, our research utilizes 1024-sized BERT embeddings as an image of size (32,32).

Previous works such as [1,6,9], use traditional Machine Learning algorithms for classification on datasets of limited scope - comparison between human-authors or consider just a single LLM. [14] discusses the application of Convolutional Neural Networks (CNNs) in character-level signal processing, serving as a motivation for our work.

The studies [11] and [16] explore the detection and regulation of AI-generated text, particularly in the context of scientific writing. In [11], a dual approach is introduced, employing feature-based methods to categorize aspects like writing style and coherence, alongside neural network-based fine-tuning of a GPT-2 output detector model using RoBERTa, achieving a 94.6% F1 score. Conversely, [16] provides a broader overview of detection techniques, encompassing black-box methods relying on API-level access and deep learning approaches involving fine-tuning LLMs like RoBERTa. The latter study also discusses white box methods, including post-hoc rule-based and neural-based approaches, as well as inference-time watermarking techniques for modifying word selection during text generation. Together, these studies contribute to understanding and regulating AI-generated text in scientific writing, offering insights into both specific detection methodologies and broader frameworks for control and regulation.

3 Dataset

In this research, we consider two datasets - Human vs LLM text [4] and Deepfake text detection [8]. In this paper, we are focused on solving a binary classification task, and a highly imbalanced dataset will produced biased results. **Hence, we balance the dataset by randomly sampling n texts from human-generated texts, where n is equal to the number of LLM-generated (GPT/Llama/FLAN/Mistral/OPT) texts in consideration.**

The Human vs LLM text dataset is a compilation of texts generated from 63 different LLMs. From the 63 LLMs, we pick texts generated from 5 different LLMs for our experiments and group variants of each LLM into a single category as in Table 1. We ensure that our dataset is always balanced.

Table 1. Distribution of Kaggle's Human vs Text corpus.

Source	Variants	No. of Samples	Min. word count per sample	Max. word count per sample
Human	-	347,692	25	71,543
FLAN	FLAN-T5-Base FLAN-T5-Large FLAN-T5-Small FLAN-T5-XXL FLAN-T5-XL	45,608	25	905
GPT	GPT-3.5 GPT-4 GPT-J GPT-NeoX	75,599	25	3,565
Llama	Llama-30B Llama-65B Llama-13B Llama-7B Llama-2-70B Llama-2-7B	42,623	25	1,770
OPT	OPT-1.3B OPT-30B OPT-2.7B OPT-6.7B OPT-125M OPT-350M OPT-13B	80,151	25	1,044
Mistral	Mistral-7B Mistral-7B-OpenOrca	10,813	25	21,734

The Deepfake text detection dataset was created by considering 10 datasets covering a wide range of writing tasks (e.g., story generation, news writing, and scientific writing) from diverse sources (e.g., Reddit posts and BBC news). 27 LLMs were employed for the construction of deepfake texts, resulting in a dataset of 447,674 instances in total. The dataset also comprised of domain-specific data from various domains, including opinion statements, news article writing, question answering, story generation, commonsense reasoning, knowledge illustration, and scientific writing.

For this study, we only considered knowledge illustration generated using 1000 Wikipedia paragraphs from the SQuAD context. Among the 27 LLMs, we have considered FLAN, GPT, Llama and OPT in this study. We have consolidated the text outputs from various sizes of the aforementioned LLMs, such as Llama 7B, 13B, 30B, and 65B, into a single category labeled "Llama". The SQuAD domain consisted of 20,950 human-generated texts and 26,714 machine-generated texts. We split this into a balanced set with different types of LLMs and an equal amount of human-authored texts, shown in Table 2.

Table 2. Distribution of Deepfake text Detection for the SQuAD domain.

Source	Variants	No. of Samples	Average Document Length	Average Sentence Length	Average # Sentences per Document
Human	-	6477	232.02	18.90	13.48
FLAN	FLAN-T5-Base FLAN-T5-Large FLAN-T5-Small FLAN-T5-XXL FLAN-T5-XL	3885	279.99	18.80	15.33
GPT	GPT-3.5 GPT-J GPT-NeoX	1830	279.99	18.80	15.33
Llama	Llama-7B Llama-13B Llama-30B Llama-65B	3102	279.99	18.80	15.33
OPT	OPT-125M OPT-350M OPT-1.3B OPT-2.7B OPT-6.7B OPT-13B OPT-30B OPT-IML-1.3B OPT-IML-30B	6477	279.99	18.80	15.33

4 Proposed Methodology

After generating the text embeddings, we reshape the 1D embedding vectors into a 2D representation before proceeding with the classification task. This reshaping is aimed at enhancing the capture of inter-embedding relationships. The neural network architecture comprises of 3 CNN layers followed by 1 fully connected layer, as depicted in Fig. 2. During training, we employ a batch size of 256, a learning rate of 0.001, and utilize the Adam optimizer.

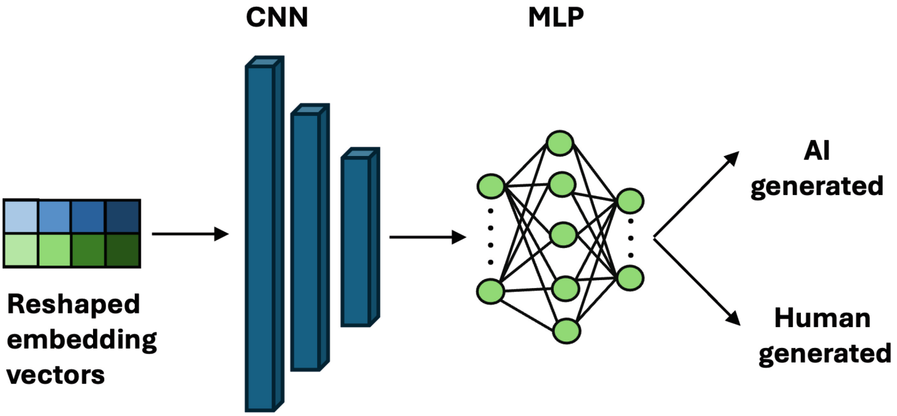


Fig. 2. Architecture of the classification model used in our framework.

4.1 Autoregressive Language Models

An Autoregressive Language Model, as in Fig. 3, is trained to predict the succeeding word x_i based on all preceding words $x_1, x_2, ..., x_{i-1}$. The training objective involves maximizing the log-likelihood $\sum_i \log(P(x_i|x_1, x_2, ..., x_{i-1}; \theta_T))$, where θ_T represents the model parameters. In Transformer decoders, these parameters reside across multiple layers of multi-head self-attention modules. Well-known models that adhere to this architecture include GPT2 [13] and Llama2 [17]. The semantic information captured by GPT2 revolves around the generation of coherent and contextually appropriate text sequences. Its training involves learning the relationships between tokens in a given context and leveraging this knowledge to generate text that follows the expected language patterns.

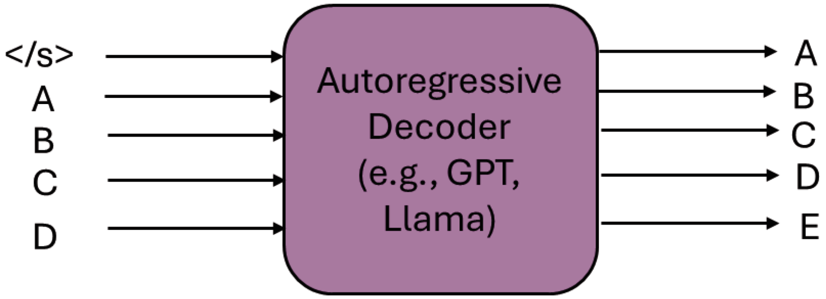


Fig. 3. Autoregressive Language Models use the provided context to generate relevant responses. Here, we assume that the token 'C' captures the context of the query.

4.2 Masked Language Models

In contrast to ALMs, Masked Language Models (MLMs) as in Fig. 4 predict a "masked" word conditioned on all other words in the sequence. During MLM training, words are randomly selected to be masked, represented by a special token [MASK], or substituted with a random token. This strategy compels the model to gather bidirectional context for making predictions. The training objective is to accurately predict the original tokens at the masked positions, expressed as $\sum_i m_i \log(P(x_i|x_1,...,x_{i-1},x_{i+1},...,x_n);\theta_T)$, where $m_i \in \{0,1\}$ denotes whether x_i is masked or not, and θ_T represents the parameters in a Transformer encoder. Notably, in models like BERT, it's common practice to mask multiple words simultaneously to facilitate parallel training. Prominent examples of MLMs include BERT [5] and RoBERTa [10]. RoBERTa [10], an enhanced framework of BERT, excels in capturing both semantic meaning and stylometric features from textual data. These embeddings can be further utilized in various downstream tasks, such as authorship attribution, by feeding them into classification models [15].

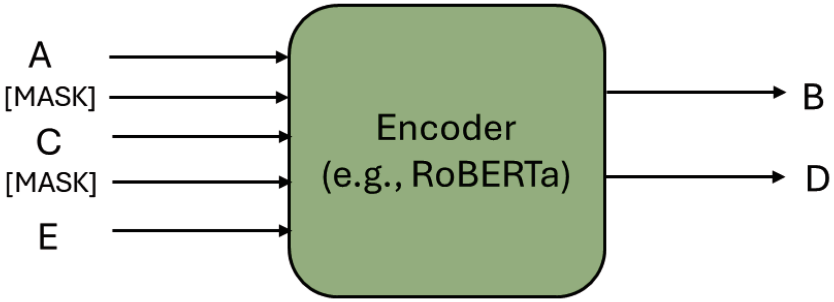


Fig. 4. Masked Language Models are trained to predict masked tokens (represented as [MASK]) in a query, which helps the models better understand semantic context.

4.3 Encoder-Decoder Language Models

The Encoder-Decoder model, as in Fig. 5, serves as a versatile "text in, text out" architecture, proficient in generating a sequence of tokens $y_1,...,y_n$ based on an input sequence $x_1,...,x_m$. When presented with a sequence pair, the training objective revolves around maximizing the log-likelihood of $\log(P(y_1,...,y_n|x_1,...,x_m);\theta_T)$, where θ_T represents the parameters within a complete encoder-decoder transformer model. To enrich the dataset for self-supervised pre-training, researchers explore various methods of sequence manipulation. These techniques involve altering the input token sequence in specific manners, with the objective of reconstructing the original sequence as the output. Examples of sequence corruption techniques include document rotation,

Table 3. Experiments on the Human vs LLM dataset assessed the accuracies of various techniques, discerning LLM ('Source') texts solely against human-authored texts.

Source	Embedding Model			Accuracy (%)	TPR	FPR	MCC
	Encoder	Decoder	Encoder-Decoder				
FLAN	RoBERTa	-	-	94.75	0.923	0.033	0.891
	-	GPT2	-	93.06	0.972	0.111	0.874
	-	Llama	-	96.10	0.983	0.061	0.923
	-	-	FLAN	97.02	0.973	0.032	0.940
	RoBERTa	GPT2	-	95.55	0.980	0.069	0.912
	RoBERTa	Llama	-	97.09	0.979	0.038	0.942
	RoBERTa	GPT2	FLAN	97.59	0.986	0.034	0.952
	RoBERTa	Llama	FLAN	97.29	0.968	0.022	0.946
	RoBERTa	**-**	**FLAN**	**97.77**	**0.978**	**0.022**	**0.955**
	-	GPT2	FLAN	97.40	0.972	0.024	0.948
	-	Llama	FLAN	96.47	0.993	0.064	0.931
GPT	RoBERTa	-	-	93.78	0.934	0.058	0.875
	-	GPT2	-	93.80	0.919	0.043	0.877
	-	Llama	-	96.51	0.959	0.028	0.930
	-	-	FLAN	98.14	0.980	0.017	0.963
	RoBERTa	GPT2	-	97.23	0.975	0.031	0.945
	RoBERTa	Llama	-	95.93	0.993	0.075	0.921
	RoBERTa	GPT2	FLAN	98.23	0.978	0.014	0.965
	RoBERTa	Llama	FLAN	97.87	0.979	0.022	0.957
	RoBERTa	**-**	**FLAN**	**98.73**	**0.987**	**0.012**	**0.975**
	-	GPT2	FLAN	98.25	0.970	0.005	0.965
	-	Llama	FLAN	97.17	0.956	0.012	0.944
Llama	RoBERTa	-	-	90.68	0.861	0.047	0.817
	-	GPT2	-	91.40	0.912	0.084	0.828
	-	Llama	-	93.50	0.951	0.082	0.870
	-	-	FLAN	94.92	0.961	0.063	0.899
	RoBERTa	GPT2	-	94.57	0.944	0.053	0.891
	RoBERTa	Llama	-	95.61	0.952	0.040	0.912
	RoBERTa	GPT2	FLAN	93.10	0.972	0.111	0.865
	RoBERTa	Llama	FLAN	95.38	0.970	0.062	0.908
	RoBERTa	**-**	**FLAN**	**96.53**	**0.949**	**0.018**	**0.931**
	-	GPT2	FLAN	95.62	0.973	0.061	0.913
	-	Llama	FLAN	95.22	0.941	0.037	0.905
OPT	RoBERTa	-	-	94.32	0.916	0.030	0.887
	-	GPT2	-	90.42	0.909	0.104	0.805
	-	Llama	-	95.86	0.970	0.053	0.917
	-	-	FLAN	97.68	0.978	0.024	0.954
	RoBERTa	GPT2	-	95.59	0.982	0.071	0.913
	RoBERTa	Llama	-	97.13	0.962	0.020	0.943
	RoBERTa	GPT2	FLAN	97.09	0.979	0.038	0.942
	RoBERTa	Llama	FLAN	97.89	0.976	0.018	0.958
	RoBERTa	**-**	**FLAN**	**98.20**	**0.978**	**0.014**	**0.964**
	-	GPT2	FLAN	97.70	0.987	0.033	0.954
	-	Llama	FLAN	96.74	0.986	0.052	0.935
Mistral	RoBERTa	-	-	99.33	0.989	0.003	0.986
	-	GPT2	-	99.42	0.996	0.007	0.988
	-	Llama	-	99.84	0.997	0.001	0.997
	-	-	FLAN	99.72	0.997	0.003	0.994
	RoBERTa	GPT2	-	99.65	0.999	0.005	0.993
	RoBERTa	Llama	-	99.91	0.999	0.000	0.998
	RoBERTa	GPT2	FLAN	99.88	0.999	0.001	0.998
	RoBERTa	Llama	FLAN	99.84	0.999	0.002	0.997
	RoBERTa	**-**	**FLAN**	**99.95**	**0.999**	**0.000**	**0.999**
	-	GPT2	FLAN	99.86	0.997	0.000	0.997
	-	Llama	FLAN	98.84	1.000	0.023	0.977

sentence permutation, text infilling, and token deletion/masking, among others [12]. FLAN [19] demonstrates proficiency in capturing semantic correlations between input and output sequences, empowering it to generate translations or summaries that maintain coherence and contextual relevance.

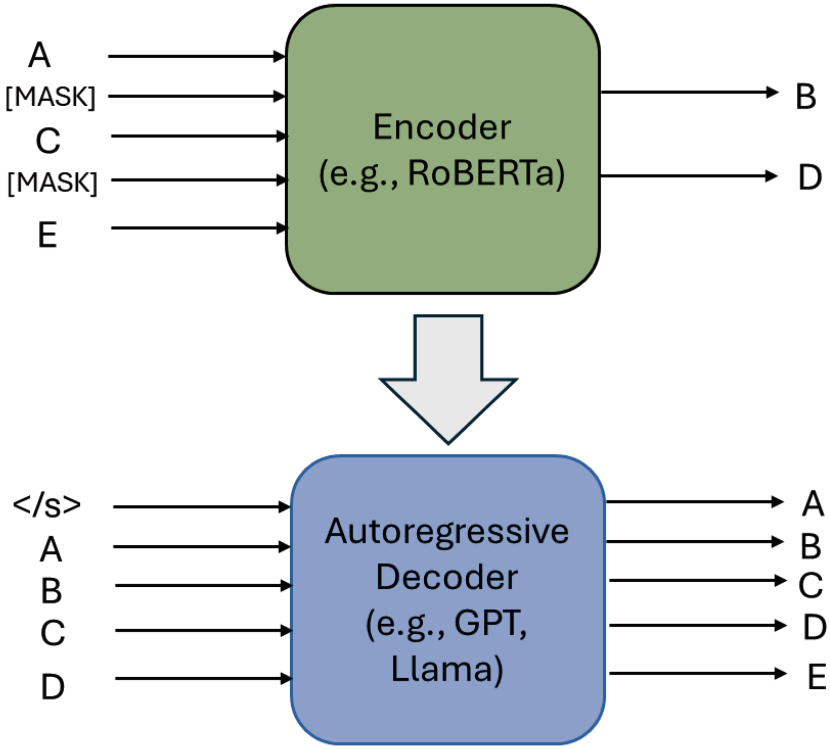


Fig. 5. Encoder-Decoder Language Models transform an input sequence into a fixed-sized embedding. These embeddings are used by the decoder to generate an output sequence. Such models are often used in machine translation and sequence-to-sequence prediction.

5 Observation and Results

From Table 3 and Table 4, we observe that the embedding combination of MLMs and EDLMs performs the best on the authorship attribution task. In addition to classification accuracy, we have also documented the performance of our approach through the Matthews Correlation Coefficient (MCC).

The equation for Matthews Correlation Coefficient (MCC) is provided in Eq. 1. MCC assesses the correlation between the actual and predicted binary

Table 4. Authorship attribution accuracies with different techniques for Deepfake text detection Dataset from the SQuAD domain.

Source	Embedding Model			Accuracy (%)	TPR	FPR	MCC
	Encoder	Decoder	Encoder-Decoder				
FLAN	RoBERTa	-	-	94.69	0.934	0.004	0.894
	-	GPT2	-	94.33	0.929	0.042	0.887
	-	-	FLAN	97.54	0.967	0.020	0.948
	-	Llama	-	98.56	0.978	0.007	0.971
	RoBERTa	GPT2	-	97.70	0.984	0.030	0.954
	RoBERTa	Llama	-	99.01	0.993	0.013	0.980
	RoBERTa	**-**	**FLAN**	**99.45**	**0.995**	**0.006**	**0.989**
	RoBERTa	GPT2	FLAN	98.87	0.993	0.017	0.976
	RoBERTa	Llama	FLAN	99.23	0.991	0.006	0.985
	-	Llama	FLAN	99.23	0.998	0.013	0.985
	-	GPT2	FLAN	99.38	0.998	0.010	0.988
GPT	RoBERTa	-	-	94.69	0.934	0.04	0.894
	-	GPT2	-	88.88	0.868	0.093	0.776
	-	-	FLAN	95.27	0.943	0.040	0.903
	-	Llama	-	88.88	0.868	0.093	0.776
	RoBERTa	GPT2	-	95.29	0.948	0.042	0.906
	RoBERTa	Llama	-	97.04	0.975	0.034	0.941
	RoBERTa	**-**	**FLAN**	**97.48**	**0.973**	**0.023**	**0.949**
	RoBERTa	GPT2	FLAN	96.34	0.964	0.038	0.925
	RoBERTa	Llama	FLAN	97.04	0.959	0.019	0.941
	-	Llama	FLAN	95.83	0.966	0.049	0.917
	-	GPT2	FLAN	96.71	0.948	0.015	0.935
Llama	RoBERTa	-	-	94.69	0.934	0.04	0.894
	-	GPT2	-	93.99	0.943	0.065	0.878
	-	-	FLAN	95.86	0.992	0.076	0.918
	-	Llama	-	94.52	0.940	0.050	0.890
	RoBERTa	GPT2	-	97.61	0.975	0.023	0.952
	RoBERTa	Llama	-	98.25	0.977	0.013	0.965
	RoBERTa	**-**	**FLAN**	**98.90**	**0.982**	**0.004**	**0.978**
	RoBERTa	GPT2	FLAN	98.79	0.982	0.008	0.974
	RoBERTa	Llama	FLAN	98.79	0.977	0.002	0.976
	-	Llama	FLAN	98.45	0.013	0.982	0.969
	-	GPT2	FLAN	98.58	0.977	0.005	0.972
OPT	RoBERTa	-	-	94.69	0.934	0.04	0.894
	-	GPT2	-	91.77	0.900	0.067	0.834
	-	-	FLAN	97.77	0.967	0.017	0.951
	-	Llama	-	97.41	0.976	0.028	0.948
	RoBERTa	GPT2	-	95.72	0.961	0.047	0.914
	RoBERTa	Llama	-	96.38	0.966	0.038	0.928
	RoBERTa	**-**	**FLAN**	**98.57**	**0.975**	**0.004**	**0.972**
	RoBERTa	GPT2	FLAN	97.33	0.964	0.019	0.945
	RoBERTa	Llama	FLAN	97.15	0.954	0.013	0.943
	-	Llama	FLAN	97.79	0.975	0.020	0.956
	-	GPT2	FLAN	98.10	0.977	0.015	0.962

classifications, with values ranging from −1 to +1. A score of +1 indicates perfect prediction, 0 denotes no correlation, and −1 implies total disagreement between predictions and actual labels.

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{1}$$

TP – True Positives TN – True Negatives

FP – False Positives FN – False Negatives

Although classification accuracy is a common metric for assessing performance in binary classification tasks, it can sometimes be misleading. This is because the train and test split can introduce small biases, even in balanced datasets. Therefore, the MCC provides a more comprehensive view of performance as it accounts for all metrics: true positives, true negatives, false positives, and false negatives. Consequently, even a 0.01 increase in the MCC scores indicates a substantial improvement in performance.

As demonstrated in Table 3 and Table 4, we notice that combining embeddings does not always enhance performance metrics. Incorporating embeddings from ALMs leads to a decrease in classification accuracy and MCC scores. This can be attributed to the nature of ALMs, which prioritise text generation and consequently assign more significance to the final tokens in the input text.

6 Conclusion and Future Work

This paper presents a pioneering framework that employs Embedding Fusion to address the longstanding challenge of distinguishing between AI-generated and human-authored text. Our approach integrates embeddings from Masked Language Models (MLMs) and Encoder-Decoder Language Models (EDLMs), concatenating them into a single feature vector. This vector is subsequently reshaped into a 2D representation to enhance the capture of inter-embedding relationships. Through extensive experimentation across 2 datasets, we achieve an accuracy >96% and Matthews Correlation Coefficient (MCC) score >0.93 showcases its effectiveness. Moreover, our findings indicate that incorporating embeddings from Autoregressive Language Models (ALMs) can degrade the information within the feature vector. We believe that the embedding fusion methodology holds significant potential for advancing authorship attribution tasks, with opportunities for further exploration through attention mechanisms and interleaving strategies.

References

1. Alamleh, H., AlQahtani, A.A.S., ElSaid, A.: Distinguishing human-written and chatgpt-generated text using machine learning. In: 2023 Systems and Information Engineering Design Symposium (SIEDS), pp. 154–158 (2023). https://doi.org/10.1109/SIEDS58326.2023.10137767
2. Dible, M.: Schools ban chatGPT amid fears of artificial intelligence-assisted cheating. Voice Am. (2023)

3. Fabien, M., Villatoro-Tello, E., Motlicek, P., Parida, S.: BertAA : BERT fine-tuning for authorship attribution. In: Bhattacharyya, P., Sharma, D.M., Sangal, R. (eds.) Proceedings of the 17th International Conference on Natural Language Processing (ICON), pp. 127–137. NLP Association of India (NLPAI), Indian Institute of Technology Patna, Patna, India (2020). https://aclanthology.org/2020.icon-main.16
4. Grinberg, Z.: Human vs. LLM text corpus (2024). https://doi.org/10.34740/KAGGLE/DSV/7378735. https://www.kaggle.com/dsv/7378735
5. Huertas-Tato, J., Huertas-Garcia, A., Martin, A., Camacho, D.: Part: pre-trained authorship representation transformer (2022)
6. Islam, N., Sutradhar, D., Noor, H., Raya, J.T., Maisha, M.T., Farid, D.M.: Distinguishing human generated text from chatGPT generated text using machine learning (2023)
7. Khan, M.A., Yadav, N., Jain, M., Goyal, S.: The art of embedding fusion: optimizing hate speech detection (2023)
8. Li, Y., et al.: Deepfake text detection in the wild (2023)
9. Lichtblau, D., Stoean, C.: Authorship attribution using the chaos game representation (2018)
10. Liu, Y., et al.: Roberta: a robustly optimized BERT pretraining approach (2019)
11. Ma, Y., et al.: Ai vs. human – differentiation analysis of scientific content generation (2023)
12. Min, B., et al.: Recent advances in natural language processing via large pre-trained language models: a survey (2021)
13. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I.: Language models are unsupervised multitask learners (2019). https://api.semanticscholar.org/CorpusID:160025533
14. Ruder, S., Ghaffari, P., Breslin, J.G.: Character-level and multi-channel convolutional neural networks for large-scale authorship attribution (2016)
15. Saedi, C., Dras, M.: Siamese networks for large-scale author identification (2021)
16. Tang, R., Chuang, Y.N., Hu, X.: The science of detecting LLM-generated texts (2023)
17. Touvron, H., et al.: Llama 2: open foundation and fine-tuned chat models (2023)
18. Verma, P.: The rise of AI fake news is creating a 'misinformation superspreader'. The Washington Post (2023)
19. Wei, J., et al.: Finetuned language models are zero-shot learners (2022)
20. Yang, M.: New York city schools ban AI chatbot that writes essays and answers prompts. The Guardian (2023)

TinyLLM Efficacy in Low-Resource Language: An Experiment on Bangla Text Classification Task

Farhan Noor Dehan, Md Fahim(✉), A. K. M. Mahabubur Rahman, M. Ashraful Amin, and Amin Ahsan Ali

Center for Computational and Data Sciences, Independent University, Bangladesh, Dhaka 1229, Bangladesh
fahimcse381@gmail.com

Abstract. Delving into the realm of Bangla text analysis, our study ventures to unlock the potential of both Large and Tiny Language Models across a range of classification tasks, from deciphering sentiment to detecting sarcasm, emotion, hate speech, and fake news. In a linguistic landscape where resources are scarce, we fill a crucial gap by meticulously evaluating model performance. Our findings unveil Gemma-2B and Bangla-BERT as top performers, with Gemma-2B excelling in detecting hate speech and sarcasm, while BanglaBERT shines in sentiment analysis and emotion detection. Notably, TinyLlama emerges as a standout, showcasing exceptional prowess in fake news detection. We emphasize the importance of selecting models attuned to the intricacies of Bangla text, with Gemma-2B, TinyLlama, and BanglaBERT exhibiting notable accuracy improvements, surpassing other contenders. Furthermore, we uncover performance disparities influenced by dataset origins, with Bangla Language Models adept at capturing social media sentiments, and Large Language Models excelling in identifying misinformation and abusive language in formal sources. Our comparison with ChatGPT's zero-shot prompting underscores the necessity for advanced NLP methodologies. By spotlighting TinyLLM, we showcase the potential of advanced NLP in Bangla text classification, paving the way for broader advancements in NLP research.

Keywords: Bangla Language Models · Multilingual Language Models · Tiny Large Language Models

1 Introduction

In the realm of NLP, the landscape of text classification has evolved significantly. Traditionally, conventional machine learning algorithms were the go-to for such tasks. However, the recent surge in transformer-based models, particularly large language models, has reshaped the field [2]. While these models have

F. N. Dehan and Md. Fahim—These authors contributed Equally to this work.

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 472–487, 2025.
https://doi.org/10.1007/978-3-031-78495-8_30

predominantly been prompt-based, their utility in languages such as Bangla has been limited due to resource constraints, including a scarcity of annotated datasets, linguistic resources, and computational infrastructure. Bangla, as a low-resource language, faces challenges in terms of data availability and linguistic resources necessary for effective NLP tasks. These limitations have made fine-tuning these models challenging in languages like Bangla. Fortunately, strides have been made with the development of smaller versions of these models, often termed"tiny" models, to broaden their accessibility and applicability, even across diverse domains [27]. Despite this progress, the exploration of these models in Bangla remains relatively under-explored, creating a notable gap in understanding their performance in Bangla text classification tasks [19].

To bridge this gap, our research endeavors to analyze the efficacy of various language models, including tiny ones, in the context of Bangla text classification tasks. Specifically, we target tasks like Sarcasm Detection [3], Hate Speech Detection [20], Bangla Fake News Detection [16], and others. Preliminary observations indicate that Tiny Large Language Models (TinyLLMs) consistently outperform existing Bangla language models (BLMs) and multilingual language models (MLMs) by substantial margins, ranging from 0.1% to 15% in most cases. By delving into these investigations, we aim to provide valuable insights into the performance of contemporary NLP models in Bangla, catering to the academic community's quest for knowledge in this domain. In this research endeavor, our contributions will encompass several key aspects:

- **Implementation of Tiny Language Models:** We implement and fine-tune tiny language models for different text classification tasks in the Bangla language. This involves adapting pre-existing models or training new ones from scratch to suit the specific linguistic nuances of Bangla.
- **Analysis of Model Performance:** We undertake thorough analyses to assess the performance of TinyLLMs in comparison to other state-of-the-art transformer models frequently employed in NLP tasks. Additionally, we evaluate these models using zero-shot prompting with ChatGPT, a state-of-the-art large language model.
- **Identification of Model Suitability:** Through rigorous experimentation and evaluation, we aim to identify the most suitable models for specific text classification tasks. This involves assessing factors such as model efficiency, robustness, and generalization capabilities.

By undertaking these endeavors, we seek to contribute to the advancement of NLP research in Bangla and facilitate the development of effective solutions for text classification tasks in this language. Our research outcomes have the potential to benefit a wide range of applications, including sentiment analysis, content moderation, and information retrieval, particularly in the context of Bangla-speaking communities. Additionally, the comparison with ChatGPT's relatively mediocre performance underscores the necessity for utilizing TinyLLMs for improved classification accuracy and effectiveness.

2 Related Works

In the domain of text classification, researchers have embarked on a journey to explore various machine and deep learning models, with pre-trained models gaining significant traction in recent years. Hasan et al. (2023) [12] delved into sentiment analysis, employing a range of machine learning models alongside fine-tuned options such as BanglaBERT and XLM-Roberta. Notably, they also incorporated ChatGPT for sentiment analysis using both zero-shot and multishot approaches. Bhattacharjee et al. (2022) [4] examined different iterations of BanglaBERT, comparing them with models like XLM-Roberta and mBERT for Bangla text analysis. However, despite this exploration, the impact of TinyLLMs has remained largely overlooked in these studies. Dehan et al. [7] investigated the performance of graph-based models for Bangla text classification. Fahim et al. [10] proposed a contextual neural stemmer for Bangla and its performance for Bangla text classification problems.

Alam et al. (2021) [1] conducted a benchmarking exercise on datasets collected from various platforms for nine NLP tasks using state-of-the-art transformer-based models. Their comparative analysis extended to monolingual versus multilingual models of varying sizes. Yet, the inclusion of Tiny LLMs in their evaluation was notably absent. Our research adopts a novel approach by broadening the scope of comparison to encompass TinyLLMs across a diverse array of tasks, including sentiment analysis, sarcasm detection, fake news detection, hate speech detection, and emotion detection. Additionally, we compare these models against a prominent large language model like ChatGPT. This comprehensive analysis aims to provide a deeper understanding of TinyLLMs performance across various datasets and tasks, while also shedding light on ChatGPT's efficacy in these domains.

In a similar vein, Kabir et al. (2023) [19] explored the application of various Large Language Models (LLMs) across a spectrum of tasks, including text classification. Their investigation incorporated zero-shot evaluation for ChatGPT, LLaMA-2, and Claude-2. However, the specific examination of TinyLLMs and LLMs was lacking, and a comprehensive analysis for each individual task was not provided. Thus, our research endeavors to fill this gap by focusing on text classification within the realm of Natural Language Understanding (NLU). Furthermore, we sought to assess ChatGPT's performance across these specific tasks.

Li et al. (2023) [21] addressed the challenges encountered by Large Language Models (LLMs) in handling low-resource languages like Bangla. Despite the potential of LLMs in NLP, their effectiveness in such languages has been limited. To tackle this issue, the authors proposed an innovative approach that integrates cross-lingual retrieval with in-context learning. By strategically utilizing prompts from languages with abundant resources that are semantically similar, they empowered Multilingual Pretrained Language Models (MPLMs), particularly emphasizing the generative model BLOOMZ, to enhance their performance on Bangla-related tasks. Their comprehensive evaluation showcased that incorporating cross-lingual retrieval consistently improves MPLMs beyond their initial zero-shot performance.

Corrêa et al. (2024) [6], akin to ours, contributes to the trend of developing LLMs for low-resource contexts, with a focus on Brazilian Portuguese. They introduce the TeenyTinyLlama (TTL) models, aiming to democratize access to LLMs and foster open-source development, especially for languages facing resource constraints. However, no research has yet compared state-of-the-art transformer models with TinyLLMs. Our study aimed to examine the factors influencing the performance of these analyzed TinyLLMs and other models, thus contributing to a deeper understanding of their capabilities in text classification tasks.

3 Methodology

Our research methodology dives into examining both the esteemed TinyLLMs and prominent language models (LMs). We refined these models through two different approaches: fine-tuning LMs using conventional methods and fine-tuning TinyLLMs using LoRA and Peft techniques.

3.1 LM Fine-tuning

In our research, we utilize a LM, which we denote as $H = f_\theta(S)$, to process input sentences and extract contextual representations. Upon tokenizing an input sentence S, represented as $T = t_1, t_2, \ldots, t_n$, the LM generates contextual representations for each token by applying the function $f_\theta(S)$, resulting in a sequence denoted as $H = h_1, h_2, \ldots, h_n$. These representations encapsulate the unique meaning of each token within the context of the entire sentence.

However, for tasks such as classification, where a fixed-size representation of the entire sentence is required, we employ a two-layer Feed Forward Neural Network (FFN) on the contextual representation of [CLS] token, h_{CLS}. This network utilizes weight matrices W_1 and W_2, bias terms b_1 and b_2, and the Rectified Linear Unit (ReLU) activation function to process h_{CLS} and generate a fixed-size representation z.

$$z = W_2 \cdot (\mathrm{ReLU}(W_1 \cdot h_{[\mathrm{CLS}]} + b_1)) + b_2 \tag{1}$$

3.2 TinyLLM Fine-Tuning Using LoRA and FEFT

Traditional fine-tuning of large language models (LLMs) involves significantly modifying the pre-trained model's parameters, which can be computationally expensive and time-consuming. PEFT (Parameter-Efficient Fine-Tuning) [22] offers a solution by adapting pre-trained models to new tasks with minimal changes to the original parameters. This significantly reduces training time and memory usage compared to traditional approaches. LoRA (Low-Rank Adaptation) [17] is a specific PEFT technique that introduces a more efficient way to capture the adjustments needed for fine-tuning. Instead of directly modifying all the pre-trained parameters, LoRA utilizes a low-rank matrix. This matrix requires significantly fewer parameters to represent the task-specific adaptations, leading to substantial efficiency gains.

Let's denote the original pre-trained model parameters as W which will be frozen during training. LoRA introduces a low-rank update, denoted by ΔW, which captures the task-specific adjustments needed for fine-tuning. This low-rank update is further decomposed as the product of two trainable matrices, A and B: $\Delta W = A \times B^T$. Here, A with a shape of $d \times r$ and B with a shape of $r \times d$ have a much lower rank (denoted by r) compared to the original dimension d of the parameter matrix W. This means they require significantly fewer parameters to represent the necessary adjustments. The rows of matrix A and the columns of matrix B can be interpreted as capturing the task-specific adaptations applied to the original weight matrix W. Finally, the updated weight metrics W' with LoRA is the summation of pretrained frozen metrics W and task-specific fine-tuned metrics ΔW

$$W' = W + \Delta W = W + AB^T$$

In essence, LoRA leverages a more compact representation (the low-rank matrices A and B) to achieve fine-tuning, resulting in significant efficiency improvements compared to traditional fine-tuning methods that modify all the pre-trained parameters directly.

3.3 Experimented Models

Experimented LMs: For fine-tuning, two different types of LM models were considered i. Bangla LM and Multilingual LM

i. Bangla LM: Our investigation delves into the renowned **BanglaBERT** and its variants, acclaimed for their effectiveness in text classification tasks, utilizing contextual embeddings from meticulous multi-stage training on Bangla corpora, crucial for our study's objectives [4,24].

ii. Multilingual LM: We also analyzed the fine-tuning performance of the multilingual language model for solving Bangla text classification tasks. In this experiment, we considered, XLM-RoBERTa [5], mBERT [8], mDeBERTa [14], and mDeBERTa-V3 [13].

Experimented TinyLLMs: In our pursuit of computational efficiency without compromising performance, we delve into the realm of TinyLLMs, exploring:

i. Gemma-2B: Gemma-2B, Google's lightweight, decoder-only language model, derived from Gemini, are versatile for text generation tasks like QA and summarization, trained on 2B parameters, enabling deployment in resource-constrained environments [25]. Gemma 2B's standout feature is its dynamic sparse attention, which efficiently allocates resources to the most relevant parts of the input, enhancing overall performance. Its modular architecture also allows for flexible scaling, adapting to different task complexities seamlessly.

ii. TinyLlama: TinyLlama, versatile and compact, trained on 1.1B parameters, ensures compatibility and ease of adoption for diverse applications [27]. TinyLlama stands out for its incredibly compact design that delivers strong language understanding while using minimal resources. Its innovative layer normalization techniques ensure that performance remains robust even with limited computational power.

iii. Falcon-1.3B: Falcon, a series of causal decoder-only models trained on 1.3B parameters, emphasizes computational efficiency with features like multi-query attention and support for efficient attention variants [23]. The Falcon-1.3B excels with its efficient use of flash attention, enabling it to achieve high performance despite its smaller size. It also integrates advanced gradient checkpointing, which optimizes memory usage during training and inference.

iv. OPT-1.3B: OPT-1.3B, utilizing causal language modeling and trained on 1.3B parameters, adeptly captures comprehensive linguistic patterns [28]. OPT-1.3B is remarkable for its open, pre-trained transformer framework, designed for easy customization and fine-tuning, all while maintaining a lean and efficient model. Additionally, its adaptive learning rate scheduler helps in fine-tuning across diverse datasets with improved stability.

4 Experiment Setup

In this study, we deployed multiple model configurations for a thorough analysis and evaluation.

4.1 Dataset

We employed five unique datasets, each designed for specific tasks including sentiment analysis, sarcasm detection, fake news detection, hate speech analysis, and emotion detection.

- **SentNoB:** A dataset comprising approximately 15k Bengali comments from diverse social media platforms across 13 domains. These comments are annotated with positive, negative, or neutral sentiments. The dataset is partitioned into roughly 13k training samples and 1.5k testing and validation samples, presenting challenges due to its noisy nature [18].
- **Bangla Sarcasm Detection Dataset:** This dataset consists of over 5k comments sourced from social media, encompassing 3k non-sarcastic and 2k sarcastic comments [3].
- **BanFakeNews:** An annotated dataset of approximately 50,000 news articles, useful for developing automated fake news detection systems. It consists of around 48,000 authentic news articles and 1,000 fabricated ones [16].
- **Hate Speech Dataset:** This dataset contains approximately 3k training samples and 1k testing samples, covering various forms of hate speech across different contexts, categorized into political, personal, gender-abusive, geopolitical, and religious hate [20].

- **YouTube Comments Emotion:** An emotion dataset containing around 3k samples with 5 classes representing different emotions, such as anger/disgust, fear/surprise, joy, sadness, and none. These samples are extracted from Bangla videos on YouTube [26].

4.2 Preprocessing and Experiment Setup

The preprocessing and experiment setup for training are discussed in detail in this section. Preprocessing steps included normalizing the text using a normalizer. We use BUET-NLP normalizer[1] in our experiment.

We use the Pytroch deep learning framework for modeling and the HuggingFace library for the pre-trained models. For LM models, we employed the AdamW optimizer with a learning rate of 1×10^{-5}, a number of epochs of 10, and a batch size of 16. Dropout regularization was applied to prevent overfitting with dropout_rate = 0.1. The hyper-parameters were chosen based on papers [9,11]

In experiments involving TinyLLMs, we established a computational environment using specialized packages like peft, bitsandbytes, and accelerate. We utilized diverse TinyLLMs variants such as falcon-1.3b, TinyLlama-1.1b, opt-1.3b and gemma-2b. For these models, we employed the AdamW optimizer with a learning rate of 2e−5 and a weight decay of 0.01. The value of r = 64, LoRA_ALPHA = 32, and LoRA_DROPOUT = 0.1. LoRA was applied to the ***all-linear*** layer of the TinyLLM. In TinyLLM experiment, models were trained for 5 epochs, with batch sizes of 2, 4, and 8, depending on the dataset size.

Tokenization was performed using Huggingface AutoTokenizer, and fine-tuning was carried out using the Huggingface Trainer module. All experiments were conducted on a single Nvidia Tesla P100 GPU.

4.3 Performance Metrics

When assessing the effectiveness of language models, several key performance metrics are relied upon to provide important insights into their performance. In our evaluation, we have focused on five widely-used metrics to gain a comprehensive understanding of the model's performance.

Accuracy. Accuracy measures the proportion of correctly classified instances among all instances. It is calculated by dividing the sum of true positives (correctly predicted positive instances) and true negatives (correctly predicted negative instances) by the total number of instances.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

[1] https://github.com/csebuetnlp/normalizer.

Precision and Recall. Precision measures the proportion of true positive instances among the instances predicted as positive, and recall measures the proportion of true positive instances that were correctly predicted out of all actual positive instances. The calculations are as follows:

$$Precision = \frac{TP}{TP + FP}, \quad Recall = \frac{TP}{TP + FN} \tag{3}$$

Macro and Weighted F1 Scores. The F1 score is the harmonic mean of precision and recall. In the macro F1 score, each class is given equal weight, and the mean of these F1 scores across all classes is calculated. Weighted F1 score, on the other hand, considers the class distribution by assigning weights to each class based on their frequency in the dataset.

Macro F1 Score:

$$F1_{macro} = \frac{1}{N}\sum_{i=1}^{N} F1_i \tag{4}$$

Weighted F1 Score:

$$F1_{weighted} = \frac{\sum_{i=1}^{N} w_i \times F1_i}{\sum_{i=1}^{N} w_i} \tag{5}$$

where N is the number of classes, w_i is the weight for class i, and $F1_i$ is the F1 score for class i.

5 Result Analysis

Through rigorous experimentation, we analyzed the performance of diverse language models on Bangla text classification datasets, revealing insights into their strengths and limitations across BLMs, MLMs, TinyLLMs, and ChatGPT, with efficacy varying based on task and dataset features.

5.1 Bangla Language Models

The performance analysis across different Bangla text classification datasets in Table 1 indicates variations in model efficacy. BanglaBERT consistently outperforms BanglaBERT-Large and BanglaBERT (Sagor Sarker) across most datasets. Notably, BanglaBERT demonstrates superior accuracy and F1 scores in SentNoB, Sarcasm Detection, Hate Speech Detection, and Emotion Detection datasets, achieving an average improvement of approximately 1–3% in accuracy and F1 scores over BanglaBERT-Large.

In Hate Speech Detection, while BanglaBERT-Large surpasses BanglaBERT in weighted F1 score, accuracy, and macro F1 score by approximately 1–3% respectively. BanglaBERT and BanglaBERT-Large also outperform

Table 1. Performance Comparison of Bangla Language Models (BLMs) on Bangla Text Classification Datasets: This table displays performance metrics, including accuracy, macro F1, and weighted F1 scores, for various Bangla Language Models evaluated across different Bangla text classification datasets. BanglaBERT emerges as the top performer across most datasets, surpassing other evaluated models.

Dataset	Model	Performance Metrics		
		Accuracy	Macro F1	Weighted F1
SentNoB	BanglaBERT	**74.46**	**69.55**	**73.03**
	BanglaBERT-Large	72.82	68.87	72.05
	BanglaBERT (Sagor Sarker)	69.42	64.54	68.01
Sarcasm Detection	BanglaBERT	**95.67**	**95.51**	**95.68**
	BanglaBERT-Large	94.55	94.23	94.50
	BanglaBERT (Sagor Sarker)	90.46	90.13	90.48
HateSpeech Detection	BanglaBERT	**69.33**	41.65	65.41
	BanglaBERT-Large	66.11	58.59	**66.96**
	BanglaBERT (Sagor Sarker)	67.11	**61.43**	66.81
BanFakeNews	BanglaBERT	96.65	92.99	96.51
	BanglaBERT-Large	**97.51**	**94.69**	**97.43**
	BanglaBERT (Sagor Sarker)	96.15	91.76	96.03
Emotion Detection	BanglaBERT	**70.78**	41.26	**65.52**
	BanglaBERT-Large	68.07	**42.87**	65.08
	BanglaBERT (Sagor Sarker)	63.86	40.10	61.09

BanglaBERT (Sagor Sarker) consistently across all datasets. These results suggest that Bangla-BERT offers notable advantages over both BanglaBERT-Large and BanglaBERT (Sagor Sarker) across various Bangla text classification tasks, while BanglaBERT-Large outperforms in certain cases. The reason for BanglaBERT's superior performance lies in its enhanced ability to grasp both semantic and syntactic contexts effectively.

5.2 Multilingual Language Models

The performance of various MLMs across different Bangla text classification datasets is summarized in Table 2. In general, XLM-Roberta consistently outperforms other MLMs across most datasets. Specifically, in SentNoB, XLM-Roberta achieves the highest accuracy, macro F1 score, and weighted F1 score, surpassing other MLMs by approximately 2–9%, indicating a significant margin of improvement. These results indicate that XLM-Roberta consistently provides superior performance compared to other Multilingual Language Models across various Bangla text classification tasks. XLM-RoBERTa exhibits superior performance compared to other multilingual models due to its advanced architecture and optimized training methodology, enabling it to capture a broader range of linguistic nuances across various languages.

Table 2. Comparative Performance of Multilingual Language Models (MLMs) on Various Bangla Text Classification Datasets: This table presents performance metrics, including accuracy, macro F1, and weighted F1 scores, for different Multilingual Language Models across several Bangla text classification datasets. The evaluated models include XLM-Roberta, M-BERT, M-deBerta, and M-deBerta-V3.

Dataset	Model	Accuracy	Macro F1	Weighted F1
SentNoB	XLM-Roberta	**70.37**	**67.94**	**70.67**
	M-BERT	67.97	65.21	68.13
	M-deBerta	60.72	55.23	58.91
	M-deBerta-V3	67.28	63.80	66.75
Sarcasm Detection	XLM-Roberta	**93.60**	**93.30**	**93.30**
	M-BERT	88.68	87.92	88.52
	M-deBerta	90.22	89.90	90.25
	M-deBerta-V3	90.63	90.01	90.50
Hate Speech Detection	XLM-Roberta	**69.44**	**62.19**	**67.95**
	M-BERT	66.22	60.65	66.09
	M-deBerta	55.22	40.95	53.05
	M-deBerta-V3	60.00	41.54	58.21
BanFakeNews	XLM-Roberta	**97.65**	**94.96**	**97.57**
	M-BERT	89.27	88.81	89.26
	M-deBerta	91.95	81.74	91.43
	M-deBerta-V3	92.98	86.43	93.12
Emotion Detection	XLM-Roberta	**67.77**	**41.39**	**63.71**
	M-BERT	59.64	34.04	55.53
	M-deBerta	51.20	24.48	44.03
	M-deBerta-V3	56.63	29.83	51.74

5.3 Tiny Large Language Models

The performance analysis of TinyLLMs across various Bangla text classification datasets is presented in Table 3. Each TinyLLM was trained on a minimum of approximately 21 billion training tokens per 1 billion parameters for Bangla text [15]. Gemma-2B consistently outperforms other TinyLLMs in terms of accuracy, macro F1 score, and weighted F1 score across all datasets. In datasets such as SentNoB, Sarcasm Detection, Hate Speech Detection, and Emotion Detection, Gemma-2B achieves the highest accuracy, macro F1 score, and weighted F1 score, outperforming TinyLlama by approximately 0.50–9% respectively. Falcon-1.3B and Opt-1.3B demonstrate comparatively lower performance metrics.

However, for BanFakeNews, both Gemma-2B and TinyLlama demonstrate comparable accuracy, with TinyLlama outperforming in terms of macro F1 score and weighted F1 score. Falcon-1.3B and Opt-1.3B again fall behind in performance across all metrics. Overall, Gemma-2B consistently demonstrates superior

Table 3. Comparative Performance of Tiny Large Language Models Across Diverse Bangla Text Classification Tasks: This table highlights accuracy, macro F1, and weighted F1 scores of various models, encompassing tasks like sentiment analysis, sarcasm detection, hate speech identification, fake news detection, and emotion detection.

Dataset	Model	Performance Metrics		
		Accuracy	Macro F1	Weighted F1
SentNoB	Gemma-2B	**66.90**	**63.02**	**66.06**
	TinyLlama	66.02	58.93	63.38
	Falcon-1.3B	58.83	46.82	52.89
	Opt-1.3B	63.18	58.13	61.70
Sarcasm Detection	Gemma-2B	**96.86**	**96.72**	**96.85**
	TinyLlama	94.13	93.87	94.12
	Falcon-1.3B	80.26	77.64	79.14
	Opt-1.3B	92.41	92.14	92.43
HateSpeech Detection	Gemma-2B	**70.89**	**63.08**	**70.30**
	TinyLlama	67.78	54.60	66.13
	Falcon-1.3B	53.56	35.43	50.51
	Opt-1.3B	56.44	32.21	51.78
BanFakeNews	Gemma-2B	**97.83**	95.50	97.80
	TinyLlama	**97.83**	**95.54**	**97.81**
	Falcon-1.3B	95.26	90.98	95.39
	Opt-1.3B	92.55	84.01	92.31
Emotion Detection	Gemma-2B	**62.65**	**36.92**	**58.62**
	TinyLlama	57.83	32.50	53.25
	Falcon-1.3B	49.10	17.45	36.22
	Opt-1.3B	48.49	15.63	34.43

performance across all datasets, highlighting its efficacy as a Large Language Model for Bangla text classification tasks. The improved efficacy demonstrated by Gemma-2B and TinyLlama in processing Bangla text could be attributed to their adept utilization of specialized knowledge tailored to the task at hand.

5.4 Evaluating ChatGPT's Zero-Shot Prompting Performance

The evaluation of ChatGPT 3.5 Turbo's zero-shot prompting for Bangla text classification is outlined in Table 4. For this experiment, we looked at different tasks and categories within each dataset. These tasks involved analyzing sentiment, spotting fake news, detecting hate speech, identifying sarcasm, and recognizing emotions. The results suggest moderate performance across diverse classification tasks. Notably, the model demonstrates superior precision and recall in Sarcasm Detection compared to other tasks. However, a noticeable

Table 4. The performance of Zero-shot Prompting with ChatGPT across diverse Bangla text classification datasets is evaluated in the table, comparing test labels against each label generated by the prompt.

Dataset	Precision	Recall	Macro F1
SentNoB	56.28	49.97	44.85
Sarcasm Detection	62.17	57.59	48.65
Hate Speech Detection	54.65	50.42	46.22
BanFakeNews	46.35	48.92	46.67
Emotion Detection	39.58	37.86	33.09

decrease is evident in Emotion Detection, indicating potential constraints in grasping nuanced emotional nuances, while showing relatively better comprehension of sarcasm. Addressing these challenges may require exploring alternative prompting techniques and fine-tuning approaches to improve task-specific performance. We revised the prompt design based on Kabir et al. (2023) [19] approach to enhance its efficiency. The subsequent illustration exemplifies the prompts employed within this study:

> For the given Input [INPUT]. Now, classify the text for [TASK]. Your output should be in between *class1,class2, . . . ,class n*. Write only your response, nothing else. Don't add anything before and after your response.

6 Findings

The performance analysis presented in Table 5 underscores the varying effectiveness of models across Bangla text classification datasets. BanglaBERT showcases superior performance SentNoB and Emotion Detection tasks, outperforming other models. Nevertheless, ChatGPT's performance appears to be notably less impressive, with accuracies falling behind by substantial margins. Gemma-2B and TinyLlama exhibits superior performance in Sarcasm Detection, HateSpeech and BanFakeNews datasets.

In this study, various language models, including Bangla Language Models, Multilingual Language Models, and Large Language Models, were fine-tuned and evaluated across distinct datasets sourced from diverse online platforms. Notably, findings from Tables 1, 2, 3, 4 and 5 reveal that Bangla Language Models exhibited superior performance when tasked with datasets originating from social media platforms such as YouTube, particularly those associated with sentiment analysis and emotion recognition. Conversely, Large Language Models demonstrated exceptional efficacy when confronted with datasets sourced from formal sources like newspapers or online articles, notably excelling in tasks such as sarcasm detection, fake news detection, and hate speech identification.

These findings indicate that different language models have varying strengths depending on the dataset's nature and origin. BLMs are sensitive to nuances in

Table 5. The table presents a performance comparison of top models across various Bangla text classification datasets, evaluating and contrasting the effectiveness of the best-performing models from BLM, MLM, TinyLLM, and ChatGPT classifications.

Dataset	Model	Performance Metrics		
		Accuracy	Macro F1	Weighted F1
SentNoB	Gemma-2B	66.90	63.02	66.06
	XLM-Roberta	70.37	67.94	70.67
	BanglaBERT	**74.46**	**69.55**	**73.03**
	ChatGPT (Zero-shot)	56.31	44.85	50.56
Sarcasm Detection	Gemma-2B	**96.86**	**96.72**	**96.85**
	XLM-Roberta	93.60	93.30	93.30
	BanglaBERT	95.67	95.51	95.68
	ChatGPT (Zero-shot)	51.10	48.65	46.46
HateSpeech Detection	Gemma-2B	**70.89**	**63.08**	**70.30**
	XLM-Roberta	69.44	62.19	67.95
	BanglaBERT	69.33	41.65	65.41
	ChatGPT (Zero-shot)	49.67	46.22	49.27
BanFakeNews	TinyLlama	**97.83**	**95.54**	**97.81**
	XLM-Roberta	97.65	94.96	97.57
	BanglaBERT-Large	97.51	94.69	97.43
	ChatGPT (Zero-shot)	82.25	46.67	77.60
Emotion Detection	Gemma-2B	62.65	36.92	58.62
	XLM-Roberta	67.77	**41.39**	63.71
	BanglaBERT	**70.78**	41.26	**65.52**
	ChatGPT (Zero-shot)	44.88	33.09	43.06

sentiment, and emotions prevalent in user-generated content on social media. In contrast, TinyLLMs are proficient in identifying patterns of misinformation and abusive language in structured, formal sources. The study highlights the significance of dataset characteristics in influencing model performance. Social media discourse, with its complex linguistic phenomena, poses challenges for TinyLLMs, resulting in lower performance compared to models fine-tuned on datasets tailored to such complexities. Conversely, the structured nature of formal text sources aligns well with the capabilities of TinyLLMs, leading to higher accuracy in tasks involving misinformation, sarcasm, and hate speech detection.

7 Conclusion

The examination of diverse language models in Bangla text classification tasks provides valuable insights into their effectiveness and applicability. Gemma-2B consistently excels in tasks like sarcasm detection and hate speech identification, showcasing its reliability and versatility. Conversely, TinyLlama stands out in fake news detection, underscoring the efficacy of specialized models in capturing subtle nuances within Bangla text. BanglaBERT demonstrated exceptional performance in the remaining selected tasks. When comparing TinyLLM's results with those of multilingual models such as XLM-Roberta and language-specific models like BanglaBERT, competitive outcomes were observed across various tasks. BLM's excel in capturing sentiment and emotions from social media, while TinyLLM's demonstrate superior capabilities in detecting sarcasm, hate speech, and fake news from formal sources.

Selecting the most suitable language model depends on factors like the task, dataset characteristics, and linguistic nuances. While Gemma-2B and TinyLlama demonstrate robust performance, XLM-Roberta, and BanglaBERT also yield commendable results. These findings offer insights for employing language models in Bangla text classification, aiding the development of accurate NLP solutions. Ongoing research is crucial to refine language models for enhanced performance and applicability in real-world scenarios.

Future Work: In our study on Bangla text classification, we faced challenges including limited annotated datasets, computational resource constraints, and potential biases in dataset characteristics. Future research could focus on expanding annotated datasets, optimizing Bangla language models, and exploring new architectures. Further analysis in specific domains, improvements in evaluation metrics, and addressing ethical concerns are also crucial. Deploying models in real-world applications and conducting user studies would provide insights into usability and effectiveness, driving further progress in the field.

Acknowledgments. We are thankful to Independent University, Bangladesh, for their support of this project. We would also like to express our gratitude to the Center for Computational & Data Sciences (CCDS Lab) for providing computational facilities and supervising this project.

References

1. Alam, F., et al.: A review of Bangla natural language processing tasks and the utility of transformer models. arXiv preprint arXiv:2107.03844 (2021)
2. Alam, T., Khan, A., Alam, F.: Bangla text classification using transformers. CoRR arXiv:2011.04446 (2020). https://arxiv.org/abs/2011.04446
3. Apon, T.S., Anan, R., Modhu, E.A., Suter, A., Sneha, I.J., Alam, M.G.R.: Banglasarc: a dataset for sarcasm detection. In: 2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE), pp. 1–5. IEEE (2022)
4. Bhattacharjee, A., et al.: BanglaBERT: language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla. In: Carpuat, M., de Marneffe, M.C., Meza Ruiz, I.V. (eds.) Findings of the Association for Computational Linguistics: NAACL 2022, pp. 1318–1327. Association for Computational Linguistics, Seattle, United States (2022). https://doi.org/10.18653/v1/2022.findings-naacl.98, https://aclanthology.org/2022.findings-naacl.98

5. Conneau, A., et al.: Unsupervised cross-lingual representation learning at scale. In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J. (eds.) Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8440–8451. Association for Computational Linguistics, Online (2020). https://doi.org/10.18653/v1/2020.acl-main.747, https://aclanthology.org/2020.acl-main.747
6. Corrêa, N.K., Falk, S., Fatimah, S., Sen, A., de Oliveira, N.: Teenytinyllama: open-source tiny language models trained in Brazilian Portuguese (2024)
7. Dehan, F., Fahim, M., Ali, A.A., Amin, M.A., Rahman, A.: Investigating the effectiveness of graph-based algorithm for bangla text classification. In: Proceedings of the First Workshop on Bangla Language Processing (BLP-2023), pp. 104–116 (2023)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (2019). https://doi.org/10.18653/v1/N19-1423, https://aclanthology.org/N19-1423
9. Fahim, M.: Aambela at blp-2023 task 2: Enhancing banglabert performance for bangla sentiment analysis task with in task pretraining and adversarial weight perturbation. In: Proceedings of the First Workshop on Bangla Language Processing (BLP-2023), pp. 317–323 (2023)
10. Fahim, M., Ali, A.A., Amin, M.A., Rahman, A.: Contextual Bangla neural stemmer: Finding contextualized root word representations for Bangla words. In: Proceedings of the First Workshop on Bangla Language Processing (BLP-2023), pp. 94–103 (2023)
11. Fahim, M., Ali, A.A., Amin, M.A., Rahman, A.M.: Edal: entropy based dynamic attention loss for hatespeech classification. In: Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation, pp. 775–785 (2023)
12. Hasan, M.A., Das, S., Anjum, A., Alam, F., Anjum, A., Sarker, A., Noori, S.R.H.: Zero-and few-shot prompting with LLMS: a comparative study with fine-tuned models for bangla sentiment analysis. arXiv preprint arXiv:2308.10783 (2023)
13. He, P., Gao, J., Chen, W.: Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing (2021)
14. He, P., Liu, X., Gao, J., Chen, W.: Deberta: decoding-enhanced BERT with disentangled attention. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=XPZIaotutsD
15. Hoffmann, J., et al.: Training compute-optimal large language models (2022)
16. Hossain, M.Z., Rahman, M.A., Islam, M.S., Kar, S.: BanFakeNews: a dataset for detecting fake news in Bangla. In: Proceedings of the Twelfth Language Resources and Evaluation Conference, pp. 2862–2871. European Language Resources Association, Marseille, France (2020). https://aclanthology.org/2020.lrec-1.349
17. Hu, E.J., et al.: Lora: low-rank adaptation of large language models (2021)
18. Islam, K.I., Kar, S., Islam, M.S., Amin, M.R.: Sentnob: a dataset for analysing sentiment on noisy bangla texts. In: Findings of the Association for Computational Linguistics: EMNLP 2021, pp. 3265–3271 (2021)
19. Kabir, M., Islam, M.S., Laskar, M.T.R., Nayeem, M.T., Bari, M.S., Hoque, E.: Benllmeval: a comprehensive evaluation into the potentials and pitfalls of large language models on bengali nlp. arXiv preprint arXiv:2309.13173 (2023)

20. Karim, M.R., Chakravarthi, B.R., Arcan, M., McCrae, J.P., Cochez, M.: Classification benchmarks for under-resourced bengali language based on multichannel convolutional-lstm network. 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), pp. 390–399 (2020). https://api.semanticscholar.org/CorpusID:215786049
21. Li, X., Nie, E., Liang, S.: Crosslingual retrieval augmented in-context learning for Bangla. In: Alam, F., Kar, S., Chowdhury, S.A., Sadeque, F., Amin, R. (eds.) Proceedings of the First Workshop on Bangla Language Processing (BLP-2023), pp. 136–151. Association for Computational Linguistics, Singapore (2023). https://doi.org/10.18653/v1/2023.banglalp-1.15, https://aclanthology.org/2023.banglalp-1.15
22. Mangrulkar, S., Gugger, S., Debut, L., Belkada, Y., Paul, S., Bossan, B.: Peft: state-of-the-art parameter-efficient fine-tuning methods (2022). https://github.com/huggingface/peft
23. Penedo, G., et al.: The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. arXiv preprint arXiv:2306.01116 (2023). https://arxiv.org/abs/2306.01116
24. Sarker, S.: Banglabert: Bengali mask language model for bengali language understanding (2020). https://github.com/sagorbrur/bangla-bert
25. Team, G.: Gemma: open models based on gemini research and technology (2024)
26. Trinto, N.I., Ali, M.E.: Detecting multilabel sentiment and emotions from bangla youtube comments. In: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1–6 (2018). https://api.semanticscholar.org/CorpusID:54440144
27. Zhang, P., Zeng, G., Wang, T., Lu, W.: Tinyllama: an open-source small language model (2024)
28. Zhang, S., et al.: OPT: open pre-trained transformer language models (2022)

Author Index

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15319, pp. 489–490, 2025.
https://doi.org/10.1007/978-3-031-78495-8