

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

LNCS 15323

Pattern Recognition

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

23 Part XXIII



Lecture Notes in Computer Science

15323

Founding Editors

Gerhard Goos
Juris Hartmanis

Editorial Board Members

Elisa Bertino, *Purdue University, West Lafayette, IN, USA*

Wen Gao, *Peking University, Beijing, China*

Bernhard Steffen , *TU Dortmund University, Dortmund, Germany*

Moti Yung , *Columbia University, New York, NY, USA*

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 XXIII

Editors

Apostolos Antonacopoulos 
University of Salford
Salford, UK

Rama Chellappa 
Johns Hopkins University
Baltimore, MD, USA

Saumik Bhattacharya 
IIT Kharagpur
Kharagpur, India

Subhasis Chaudhuri 
Indian Institute of Technology Bombay
Mumbai, India

Cheng-Lin Liu 
Chinese Academy of Sciences
Beijing, China

Umapada Pal 
Indian Statistical Institute Kolkata
Kolkata, India

ISSN 0302-9743

ISSN 1611-3349 (electronic)

Lecture Notes in Computer Science

ISBN 978-3-031-78346-3

ISBN 978-3-031-78347-0 (eBook)

<https://doi.org/10.1007/978-3-031-78347-0>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2025

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

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

If disposing of this product, please recycle the paper.

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 bio-signal 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 IJCPD 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
Josef Kittler
Anil Jain

Indian Statistical Institute, Kolkata, India
University of Surrey, UK
Michigan State University, USA

Program Chairs

Apostolos Antonacopoulos
Subhasis Chaudhuri
Rama Chellappa
Cheng-Lin Liu

University of Salford, UK
Indian Institute of Technology, Bombay, India
Johns Hopkins University, USA
Institute of Automation, Chinese Academy of
Sciences, China

Publication Chairs

Ananda S. Chowdhury
Wataru Ohyama

Jadavpur University, India
Tokyo Denki University, Japan

Competition Chairs

Richard Zanibbi
Lianwen Jin
Laurence Likforman-Sulem

Rochester Institute of Technology, USA
South China University of Technology, China
Télécom Paris, France

Workshop Chairs

P. Shivakumara
Stephanie Schuckers
Jean-Marc Ogier
Prabir Bhattacharya

University of Salford, UK
Clarkson University, USA
Université de la Rochelle, France
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 Lladós	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	Da-Han Wang
Mingxin Huang	Yang Xue
Lei Kang	Wentao Yang
Wenhui Liao	Jiixin Zhang
Yuliang Liu	Yiwu Zhong
Yongxin Shi	

Reviewers (Conference Papers)

Aakanksha Aakanksha
 Aayush Singla
 Abdul Muqet
 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	Bin-Bin Jia
Arnav Varma	Binbin Yong
Arpita Dutta	Bindita Chaudhuri
Arshad Jamal	Bindu Madhavi Tummala
Artur Jordao	Binh M. Le
Arunkumar Chinnaswamy	Bi-Ru Dai
Aryan Jadon	Bo Huang
Aryaz Baradarani	Bo Jiang
Ashima Anand	Bob Zhang
Ashis Dhara	Bowen Liu
Ashish Phophalia	Bowen Zhang
Ashok K. Bhateja	Boyang Zhang
Ashutosh Vaish	Boyu Diao
Ashwani Kumar	Boyun Li
Asifuzzaman Lasker	Brian M. Sadler
Atefeh Khoshkhahtinat	Bruce A. Maxwell
Athira Nambiar	Bryan Bo Cao
Attilio Fiandrotti	Buddhika L. Semage
Avandra S. Hemachandra	Bushra Jalil
Avik Hati	Byeong-Seok Shin
Avinash Sharma	Byung-Gyu Kim
B. H. Shekar	Caihua Liu
B. Uma Shankar	Cairong Zhao
Bala Krishna Thunakala	Camille Kurtz
Balaji Tk	Carlos A. Caetano
Balázs Pálffy	Carlos D. Martá-Nez-Hinarejos
Banafsheh Adami	Ce Wang
Bang-Dang Pham	Cevahir Cigla
Baochang Zhang	Chakravarthy Bhagvati
Baodi Liu	Chandrakanth Vipparla
Bashirul Azam Biswas	Changchun Zhang
Beiduo Chen	Changde Du
Benedikt Kottler	Changkun Ye
Beomseok Oh	Changxu Cheng
Berkay Aydin	Chao Fan
Berlin S. Shaheema	Chao Guo
Bertrand Kerautret	Chao Qu
Bettina Finzel	Chao Wen
Bhavana Singh	Chayan Halder
Bibhas C. Dhara	Che-Jui Chang
Bilge Günsel	Chen Feng
Bin Chen	Chenan Wang
Bin Li	Cheng Yu
Bin Liu	Chenghao Qian
Bin Yao	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 Coquenot
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
Dipanjoyoti 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 Binamakhshen
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 Otroschi 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
Jiyang 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
Liviú-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 Kämpel
Martina Pastorino
Marwan Turki
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	Palash Ghosal
Narayan Vetrekar	Pallav Dutta
Narendra D. Londhe	Paolo Rota
Nathalie Girard	Paramanand Chandramouli
Nati Ofir	Paria Mehrani
Naval Kishore Mehta	Parth Agrawal
Nazmul Shahadat	Partha Basuchowdhuri
Neeti Narayan	Patrick Horain
Neha Bhargava	Pavan Kumar
Nemanja Djuric	Pavan Kumar Anasosalu Vasu
Newlin Shebiah R.	Pedro Castro
Ngo Ba Hung	Peipei Li
Nhat-Tan Bui	Peipei Yang
Niaz Ahmad	Peisong Shen
Nick Theisen	Peiyu Li
Nicolas Passat	Peng Li
Nicolas Ragot	Pengfei He
Nicolas Sidere	Pengrui Quan
Nikolaos Mitianoudis	Pengxin Zeng
Nikolas Ebert	Pengyu Yan
Nilah Ravi Nair	Peter Eisert
Nilesh A. Ahuja	Petra Gomez-Krämer
Nilkanta Sahu	Pierrick Bruneau
Nils Murrugarra-Llerena	Ping Cao
Nina S. T. Hirata	Pingping Zhang
Ninad Aithal	Pintu Kumar
Ning Xu	Pooja Kumari
Ningzhi Wang	Pooja Sahani
Niraj Kumar	Prabhu Prasad Dev
Nirmal S. Punjabi	Pradeep Kumar
Nisha Varghese	Pradeep Singh
Norio Tagawa	Pranjal Sahu
Obaidullah Md Sk	Prasun Roy
Oguzhan Ulucan	Prateek Keserwani
Olfa Mechi	Prateek Mittal
Oliver Tüselmann	Praveen Kumar Chandaliya
Orazio Pontorno	Praveen Tirupattur
Oriol Ramos Terrades	Pravin Nair
Osman Akin	Preeti Gopal
Ouadi Beya	Preety Singh
Ozge Mercanoglu Sincan	Prem Shanker Yadav
Pabitra Mitra	Prerana Mukherjee
Padmanabha Reddy Y. C. A.	Prerna A. Mishra
Palaash Agrawal	Prianka Dey
Palaiahnakote Shivakumara	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
Ruwei 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
Siladitya 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 Napol on
 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 B chner

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

How to Modify the Tree of Shapes of an Image: Connected Operators Without Gradient Inversion	1
<i>Julien Mendes Forte, Nicolas Passat, and Yukiko Kenmochi</i>	
New Algorithms for Multivalued Component Trees	19
<i>Nicolas Passat, Romain Perrin, Jimmy Francky Randrianasoa, Camille Kurtz, and Benoît Naegel</i>	
Sketch2Seg: Sketch-Based Image Segmentation with Pre-trained Diffusion Model	36
<i>Xin Dai, Haoge Deng, Ke Li, and Yonggang Qi</i>	
Unsupervised Segmentation of Pulmonary Regions in 3D CT Scans Optimized Using Transformer Model	51
<i>Ahmed Sharafeldeen, Adel Khelifi, Mohammed Ghazal, Maha Yaghi, Ali Mahmoud, Sohail Contractor, and Ayman El-Baz</i>	
Deep Spherical Superpixels	67
<i>Rémi Giraud and Michaël Clément</i>	
External Prompt Features Enhanced Parameter-Efficient Fine-Tuning for Salient Object Detection	82
<i>Wen Liang, Peipei Ran, Mengchao Bai, Xiao Liu, P. Bilha Githinji, Wei Zhao, and Peiwu Qin</i>	
Recall-Based Knowledge Distillation for Data Distribution Based Catastrophic Forgetting in Semantic Segmentation	98
<i>Samiha Mirza, Apurva Gala, Pandu Devarakota, Vuong D. Nguyen, Pranav Mantini, and Shishir K. Shah</i>	
Contrastive Gaussian Clustering for Weakly Supervised 3D Scene Segmentation	114
<i>Myrna Castillo, Mahtab Dahaghin, Matteo Toso, and Alessio Del Bue</i>	
AYANet: A Gabor Wavelet-Based and CNN-Based Double Encoder for Building Change Detection in Remote Sensing	131
<i>Priscilla Indira Osa, Josiane Zerubia, and Zoltan Kato</i>	

Task Consistent Prototyp Learning for Incremental Few-Shot Semantic Segmentation	147
<i>Wenbo Xu, Yanan Wu, Haoran Jiang, Yang Wang, Qiang Wu, and Jian Zhang</i>	
Multi-scale Value-Density Transformer with Medical Semantic Guidance for Disease Risk Prediction Based on Clinical Time Series	163
<i>Jingwen Xu, Xiaoge Wei, and Pong C. Yuen</i>	
Variational Autoencoder Learns Better Feature Representations for EEG-Based Obesity Classification	179
<i>Yuan Yue, Dirk De Ridder, Patrick Manning, and Jeremiah D. Deng</i>	
A Deep Learning System for Water Pollutant Detection Based on the SENSIPLUS Microsensor	192
<i>Hamza Mustafa, Mario Molinara, Luigi Ferrigno, and Michele Vitelli</i>	
Bandwise Attention in CycleGAN for Fructose Estimation from Hyperspectral Images	204
<i>Divyani Tyagi and Tushar Sandhan</i>	
Unsupervised Multi-level Search and Correspondence for Generic Voice-Face Feature Spaces	219
<i>Jing Sun and Jianbo Su</i>	
KunquDB: An Attempt for Speaker Verification in the Chinese Opera Scenario	233
<i>Huali Zhou, Yuke Lin, Dong Liu, and Ming Li</i>	
Act-ChatGPT: Introducing Action Features into Multi-modal Large Language Models for Video Understanding	250
<i>Yuto Nakamizo and Keiji Yanai</i>	
Machine Vision-Aware Quality Metrics for Compressed Image and Video Assessment	266
<i>Mikhail Dremin, Konstantin Kozhemyakov, Ivan Molodetskikh, Malakhov Kirill, Sagitov Artur, and Dmitriy Vatolin</i>	
Author Index	283



How to Modify the Tree of Shapes of an Image: Connected Operators Without Gradient Inversion

Julien Mendes Forte^{1(✉)}, Nicolas Passat², and Yukiko Kenmochi¹

¹ Normandie Univ, UNICAEN, ENSICAEN, CNRS, GREYC, 14050 Caen, France
julien.mendes-forte@unicaen.fr

² Université de Reims Champagne-Ardenne, CRESTIC, 51100 Reims, France

Abstract. The tree of shapes is a hierarchical data structure that models a grey-level image via its level lines. It belongs to the family of morphological trees, which allow to design connected operators, i.e. non-linear filters that transform an image without creating new contours. Connected operators act by modifying the image-modeling tree, shifting the values of its nodes. This paradigm is frequently used with the component tree, another popular morphological tree. It is much less considered in the case of the tree of shapes despite its ability to model more finely the image. Indeed, shifting the values of the nodes of a tree of shapes is more complex, compared to other morphological trees. In this article, we investigate how to modify a tree of shapes by shifting the values of its nodes. We explain how to carry out this operation so that the modified/simplified tree remains the tree of shapes of the processed image. We propose algorithmic solutions and methodological schemes to reach that goal. We discuss on their properties and we illustrate their relevance by application examples of induced connected operators. Software implementation available at <https://github.com/jmendesf/ToSConOp>.

Keywords: tree of shapes · connected filters · grey-level imaging · hierarchical models · mathematical morphology

1 Introduction

Graph-based hierarchical structures are a cornerstone of mathematical morphology. These hierarchical structures—most often trees—allow to model, describe and analyse images. They also provide a way to process images, by allowing the design of connected operators [29], a family of non-linear filters which share the specific property of modifying an image while preserving its contours (i.e. not

Work funded by Région Normandie (thesis grant RIN), Partenariats Hubert Curien (Sakura, 49674RK) and Agence Nationale de la Recherche (ANR-20-CE45-0011, ANR-22-CE45-0034, ANR-23-CE45-0015). ANATOMIX is an EQUIPEX funded by the Investments for the Future program of the Agence Nationale de la Recherche (ANR-11-EQPX-0031).

creating contours that do not exist in the input image). This property derives from the fact that these operators do not act at the scale of the pixels, but at the scale of the connected components/flat zones of the image modeled by a tree [28]. The applications of such operators are many, including mainly filtering and segmentation.

Two kinds of trees were proposed for modeling images: the component tree [27] and the tree of shapes [17]. Both were initially designed for modeling grey-level images and further generalized to handle multivalued images [7, 14]. The design of connected operators based on such trees relies on a three-step procedure [13]: (1) building the tree from the original image (2) modifying the tree (generally, simplifying it, which can be done by attribute analysis [4] or shaping [32]), and (3) reconstructing the final image from the modified tree. While component tree based connected operators enjoy a wide variety of applications, they are not well-fitted when brighter and darker areas of the images need to be handled simultaneously. In such cases, using both the min- and max-trees in a parallel or iterative fashion may lead to unsatisfactory results. Despite the strong links that exist between the component tree and the tree of shapes [21], and while connected operators based on the tree of shapes have been proposed [33], this three-step procedure was mostly used to develop connected operators based on the component tree.

This fact derives from the distinct modeling proposed by the two kinds of trees. The component tree builds upon the connected components of the binary threshold sets of the image. As a consequence, simplifying that tree, i.e. removing a node, provides a valid component tree. It also has a straightforward effect on the image, by lowering the values of the region associated to the node, leading in particular to gradient-sign preserving, anti-extensive filters. By contrast, the tree of shapes builds upon regions bounded by the level lines of the image. As a consequence, simplifying that tree by removing a node (or more generally by modifying its value) provides a tree which may not be a valid tree of shapes. It may also have unexpected effects on the image, in particular by modifying the sign of the gradients of the contours.

To the best of our knowledge, the way to modify the value of the nodes in a tree of shapes without altering its intrinsic properties nor its ability to model the associated image had not been investigated until now [10]. We propose to tackle this issue.

This article is organized as follows. Section 2 describes related works on the tree of shapes and connected operators. Section 3 provides notions and properties related to the tree of shapes. Section 4 describes how an image can be modified by shifting the value of the nodes of its tree of shapes, and discusses on the related issues. Section 5 provides an algorithm for shifting the value of one node of a tree of shapes whereas guaranteeing that it remains the tree of shapes of the modified image. Section 6 proposes a generic scheme that extends the one-node shifting algorithm as a many-node shifting. Section 7 exemplifies this scheme with induced connected operators. Section 8 concludes the article.

2 Related Works

The trees developed in mathematical morphology are subdivided into two families. The first gathers the total partition trees (e.g. the binary partition tree [26] and the watershed tree [18]) that decompose the image into incrementally refining partitions. The second gathers the partial partition trees that model images with respect to their spatial-spectral structure. The tree of shapes [17] belongs to this second family. It models a grey-level image by considering it as a topographic map. It encodes the nested relation between the isocontours of the image. It is the grey-level generalization of the adjacency tree [24] that models the topological structure of a binary image.

Modeling an image by a tree of shapes requires to fulfill specific topological constraints. Indeed, the support of the image has to satisfy the Jordan-Brouwer property. This is guaranteed for certain topological frameworks, e.g. for the well-composed images [15]. Digitally well-composed interpolations were investigated for the definition of trees of shapes in arbitrary dimensions [3].

Variants of the tree of shapes were proposed. One can cite the multivariate tree of shapes [7] that handles non-grey-level images, or the topological monotonic tree [30] and the topological tree of shapes [20], that focus on the topological structure of the tree of shapes. The links that exist between the tree of shapes and the component tree [27] (the other archetype of the partial partition trees) were also investigated. These links are known since their introduction, and were mainly characterized by the hole-filling procedure between the nodes of the component trees and those of the tree of shapes. Recently, the homeomorphic links between both trees were explicitated and formalized [21].

Efforts were geared towards efficiently building the tree of shapes. A method based on immersing the image domain in the Khalimsky grid guarantees a worst-case quasi-linear complexity [12]. Alternative approaches, including a root-to-leaves paradigm [16], have also been presented, and parallel strategies were investigated [11]. The literature on the construction of the component tree and the tree of shapes proposes some algorithms that rely on the first to build the second [5, 9] or vice versa [31].

The rich information modeled by the tree of shapes allowed the development of various image processing and analysis approaches: filtering [8], segmentation [1], simplification [2] or object recognition [19]. The image processing methods based on the tree of shapes belong to the family of the connected operators [29]. Such operators model an image via a morphological tree (tree of shapes, component tree), modify that tree, and reconstruct the image accordingly [13]. The modification of the tree is most often a simplification that discards the nodes of the tree that do not satisfy a given criterion, based on descriptive attributes [4].

Connected operators based on this paradigm mainly build upon component trees, generally leading to anti-extensive filters. This sheds light on the spectral anisotropy of these component tree-based operators. By contrast, the simplification (or more generally, the modification) of a tree of shapes generally belongs to the class of self-dual connected operators, leading to an isotropic behaviour,

relevant for many image processing applications. Such tree-of-shapes-based connected operators are especially interesting as they can take into account both minimal (“dark”) and maximal (“light”) regions of the image. However, modifying/simplifying a tree of shapes in a convenient fashion is an operation that is more complex than for a component tree (see Sect. 4).

3 Tree of Shapes

3.1 Basic Definitions

An image is defined as a function $\mathcal{F} : \mathbb{U} \rightarrow \mathbb{V}$ (Fig. 1(a)). We assume that (1) \mathbb{U} is endowed with a topological structure compliant with the Jordan-Brouwer property and (2) \mathbb{V} is endowed with a total order relation \leq . For the sake of readability, we set $\mathbb{U} = \mathbb{Z}^d$ ($d \geq 2$) endowed with the digital topology framework [25]. We also set $\mathbb{V} = \mathbb{Z}$ so that $\mathcal{F}(\mathbb{U}) = \llbracket \perp, \top \rrbracket \subset \mathbb{V}$ and a finite number of points $\mathbf{x} \in \mathbb{U}$ satisfy $\mathcal{F}(\mathbf{x}) > \perp$. These hypotheses are generally satisfied by digital images.

Let $v \in \mathbb{V}$. The upper- and lower-threshold sets of \mathcal{F} at value v (Fig. 1(b,d)) are the subsets of \mathbb{U} defined as

$$A_v^\circ(\mathcal{F}) = \{\mathbf{x} \in \mathbb{U} \mid v \leq \mathcal{F}(\mathbf{x})\} \quad \text{and} \quad A_v^\bullet(\mathcal{F}) = \{\mathbf{x} \in \mathbb{U} \mid v > \mathcal{F}(\mathbf{x})\} \quad (1)$$

Let $A \subseteq \mathbb{U}$. We note $\Pi[A] \subseteq 2^{\mathbb{U}}$ the set of the connected components of A (we set $\Pi[A] = \emptyset$ if $A = \emptyset$). We note

$$\mathcal{C}^\circ = \bigcup_{v \in \mathbb{V}} \Pi[A_v^\circ(\mathcal{F})] \quad \text{and} \quad \mathcal{C}^\bullet = \bigcup_{v \in \mathbb{V}} \Pi[A_v^\bullet(\mathcal{F})] \quad (2)$$

the set of the connected components of \mathcal{F} induced by its upper- and lower-threshold sets, respectively (Fig. 1(b,d)).

Let $Z \in \Pi[A]$. We note $Z^\tau \supseteq Z$ the connected set obtained by filling the holes of Z (Fig. 1(f)). We set $\mathcal{C} = \mathcal{C}^\circ \cup \mathcal{C}^\bullet$ and

$$\Theta = \{Z^\tau \mid Z \in \mathcal{C}\} \quad (3)$$

We consider the partial order relation \subseteq on Θ and we note \triangleleft its reflexive-transitive reduction. The tree of shapes (Fig. 1(g)) is then defined as follows.

Definition 1 (Tree of shapes [17]). *The tree of shapes of \mathcal{F} is the tree $\mathfrak{T} = (\Theta, \triangleleft)$.*

The elements of Θ (resp. \triangleleft) are called the *nodes* (resp. the *edges*) of \mathfrak{T} .

3.2 Proper Part, Altitude, Image Reconstruction

Each node $X \in \Theta$ is characterized spatially (with respect to \mathbb{U}) and spectrally (with respect to \mathbb{V}). This leads to the two notions of *proper part* and the *altitude* of a node.

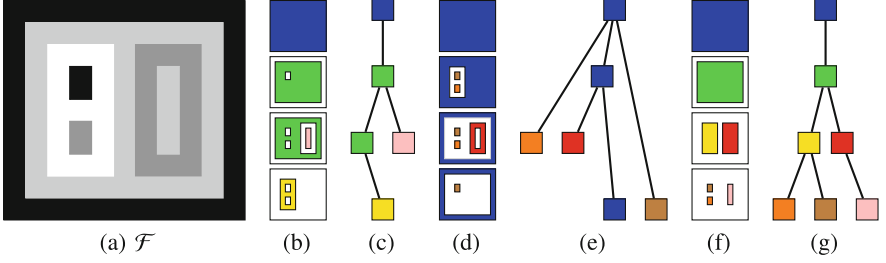


Fig. 1. An image \mathcal{F} (a), its upper- (b) and lower-threshold sets (d) (coloured regions). The two component trees of \mathcal{F} : the max-tree (c) that derives from (b) and the min-tree (e) that derives from (d). Both model the inclusion between connected components. The tree of shapes (g) that derives from the hole filling of the connected components of (b,d) of same color lead to the node of this color in (f,g).

Definition 2 (Proper part of a node). Let $X \in \Theta$. The proper part of X in the tree of shapes $\mathfrak{T} = (\Theta, \triangleleft)$ is defined by

$$\rho(X) = X \setminus \bigcup_{Y \triangleleft X} Y \quad (4)$$

We note $\rho : \Theta \rightarrow 2^{\mathbb{U}}$ the induced function.

Remark 3. The set $\{\rho(X) \mid X \in \Theta\}$ is a partition of \mathbb{U} .

Definition 4 (Altitude of a node). Let $X \in \Theta$. The altitude $Alt(X)$ of X in the tree of shapes $\mathfrak{T} = (\Theta, \triangleleft)$ is defined by

$$\forall \mathbf{x} \in \rho(X), Alt(X) = \mathcal{F}(\mathbf{x}) \quad (5)$$

The altitude $Alt(X)$ of X is the unique value of \mathbb{V} that \mathcal{F} assigns to the points of the proper part $\rho(X)$ of X . We note $Alt : \Theta \rightarrow \mathbb{V}$ the induced function.

From now on, we consider that the tree of shapes \mathfrak{T} is implicitly endowed with its induced proper part and altitude functions, i.e. we consider \mathfrak{T} as $(\mathfrak{T}, \rho, Alt)$. From $(\mathfrak{T}, \rho, Alt)$, it is possible to reconstruct the image in a lossless way.

Property 5. Given the tree of shapes $(\mathfrak{T} = (\Theta, \triangleleft), \rho, Alt)$, the image $\mathcal{F} : \mathbb{U} \rightarrow \mathbb{V}$ can be recovered by setting, for any $\mathbf{x} \in \mathbb{U}$

$$\mathcal{F}(\mathbf{x}) = \bigvee_{X \in \Theta}^{\leq} \mathbf{1}_{(\rho(X), Alt(X))}(\mathbf{x}) \quad (6)$$

where \bigvee^{\leq} is the supremum for \leq , and $\mathbf{1}_{(Y,u)} : \mathbb{U} \rightarrow \mathbb{V}$ is the cylinder function defined by $\mathbf{1}_{(Y,u)}(\mathbf{x}) = u$ if $\mathbf{x} \in Y$ and \perp otherwise.

In other words, the image \mathcal{F} can be reconstructed by assigning to each point $\mathbf{x} \in \mathbb{U}$ the altitude value $Alt(X)$ of the node $X \in \Theta$ that contains \mathbf{x} in its proper part $\rho(X)$. This reconstruction formula constitutes the cornerstone for designing connected operators based on the tree of shapes of an image, as discussed in the next section.

4 How to Modify an Image via Its Tree of Shapes?

Before discussing on the connected operators based on morphological trees (Sect. 4.2), we recall properties of the tree of shapes related to its invariance to contrast transformations (Sect. 4.1).

4.1 Contrast Invariance of the Tree of Shapes

Let $\gamma : \mathbb{V} \rightarrow \mathbb{V}$ be a transformation of the space of values. We say that γ is a contrast transformation if it is strictly increasing, i.e. if for all $u, v \in \mathbb{V}$ we have $(u < v) \Rightarrow (\gamma(u) < \gamma(v))$.

Property 6 (Contrast invariance of the tree of shapes). *Let $\mathcal{F} : \mathbb{U} \rightarrow \mathbb{V}$ be an image. Let $\gamma : \mathbb{V} \rightarrow \mathbb{V}$. Let $(\mathfrak{T}, \rho, Alt)$ be the tree of shapes of \mathcal{F} . If γ is a contrast transformation, then $(\mathfrak{T}, \rho, \gamma \circ Alt)$ is the tree of shapes of $\gamma \circ \mathcal{F}$.*

In other words, a contrast transformation γ alters neither the structural (\mathfrak{T}) nor the spatial (ρ) part of the tree of shapes of an image, whereas the spectral part ($\gamma \circ Alt$) directly follows the contrast transformation.

A contrast transformation γ acts on the space of values \mathbb{V} . It induces a contrast transformation $\Gamma : \mathbb{V}^{\mathbb{U}} \rightarrow \mathbb{V}^{\mathbb{U}}$ that acts on the space of images. It is defined, for any image $\mathcal{F} : \mathbb{U} \rightarrow \mathbb{V}$, by $\Gamma(\mathcal{F}) = \gamma \circ \mathcal{F}$. The function Γ exhibits an important property: it is gradient-sign preserving. Let $\mathbf{x}, \mathbf{y} \in \mathbb{U}$ such that \mathbf{x} and \mathbf{y} are neighbour with respect to the topological structure of \mathbb{U} . Let $\mathcal{F} : \mathbb{U} \rightarrow \mathbb{V}$ be an image. The gradient sign-preserving property of Γ means that

$$(\mathcal{F}(\mathbf{x}) \leq \mathcal{F}(\mathbf{y})) \implies (\Gamma(\mathcal{F}(\mathbf{x})) \leq \Gamma(\mathcal{F}(\mathbf{y}))) \quad (7)$$

As a corollary of Eq. (7), we have

$$(\mathcal{F}(\mathbf{x}) = \mathcal{F}(\mathbf{y})) \implies (\Gamma(\mathcal{F}(\mathbf{x})) = \Gamma(\mathcal{F}(\mathbf{y}))) \quad (8)$$

From Eq. (8), Γ is a connected operator, i.e. it does not create new contours in the transformed image.

From Properties 5 and 6, the application of Γ on the image \mathcal{F} could be carried out by (1) building the tree of shapes $(\mathfrak{T}, \rho, Alt)$ of \mathcal{F} ; (2) modifying Alt into $\gamma \circ Alt$; (3) reconstructing the image $\Gamma(\mathcal{F})$ modeled by $(\mathfrak{T}, \rho, \gamma \circ Alt)$ (see Eq. (6)).

For a trivial transformation such as Γ , acting directly on \mathcal{F} would be sufficient. Nonetheless, the three-step procedure inspired from [13] and discussed in Sect. 4.2 opens the way to the design of a wider range of gradient-sign preserving connected operators based on the modification of the values of the nodes of the tree of shapes, leading to *local* modifications of the contrast of the modeled images.

4.2 Connected Filtering Framework: Issues and Purpose

As stated above, a connected operator can be designed from the simplification of a tree.

The process consists in:

	$\mathcal{F} \xrightarrow{(i)} \mathfrak{T}$	
(i) building the tree \mathfrak{T} that models the image \mathcal{F} ;	$\downarrow \Gamma$	$\downarrow (ii)$
(ii) simplifying \mathfrak{T} into a new tree $\widehat{\mathfrak{T}}$; and	$\widehat{\mathcal{F}} \xleftarrow{(iii)} \widehat{\mathfrak{T}}$	(9)
(iii) reconstructing the new image $\widehat{\mathcal{F}}$ from $\widehat{\mathfrak{T}}$.		

It was pioneered in [13], where it was designed for component trees [27] and by considering that the simplification of Step (ii) consists of discarding nodes according to attributes [4].

This framework may be considered also with a tree of shapes instead of a component tree. Step (i) relies on the construction of the tree, which can be done either for the component tree [6] or the tree of shapes [12]. Step (iii) relies on the reconstruction formula of Eq. (6), which is similar for both the component tree and the tree of shapes.

Regarding Step (ii), in [13] and most of the subsequent contributions, the idea was to preserve some nodes of the component tree whereas discarding others. In particular, when a node X was preserved, $\rho(X)$ and $Alt(X)$ were unaltered. By contrast, when a node X was discarded, $\rho(X)$ was merged with the proper part $\rho(P)$ of its parent node P such that $X \triangleleft P$. This discarding/merging can be seen as a side effect of the modification of the altitude $Alt(X)$ so that it becomes equal to $Alt(P)$. Step (ii) could then be generalized as “turn \mathfrak{T} into a new tree $\widehat{\mathfrak{T}}$ by modifying the altitudes of its nodes”.

Defining a new altitude function $\widehat{Alt} : \Theta \rightarrow \mathbb{V}$ does not modify the structure of the tree \mathfrak{T} . Besides, the reconstruction formula of Eq. (6) remains valid, making Step (iii) tractable. The induced operator Γ applied on \mathcal{F} is a connected operator, since it satisfies Eq. (8). Nonetheless, two problems may occur (Fig. 2). First, the tree \mathfrak{T}' associated to the new altitude application \widehat{Alt} , and to the induced image $\widehat{\mathcal{F}}$ may not be the tree of shapes $\widehat{\mathfrak{T}}$ of $\widehat{\mathcal{F}}$. More generally, it may not be a tree of shapes (Fig. 2(c,e)). Second, the operator Γ , although being a connected operator, may not satisfy Eq. (7), i.e. it may not be gradient sign-preserving (Fig. 2(a,d)).

We propose hereafter a framework that allows to modify the altitude of the nodes of a tree of shapes—with controlled side effects on its structure—so that:

- (P₁) the modified tree remains the tree of shapes of the associated modified image;
- (P₂) the sign of the gradients of the modified image is preserved with respect to the initial image.

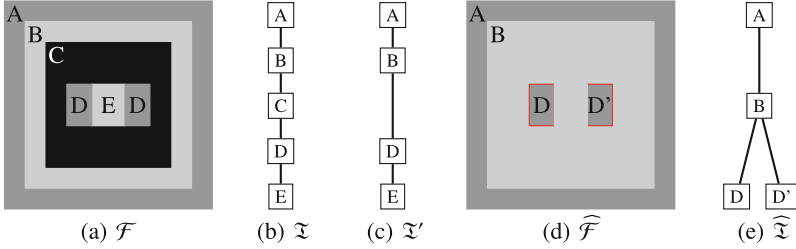


Fig. 2. An image \mathcal{F} (a) and its tree of shapes \mathfrak{T} (b). The removal of C (equivalent to setting its altitude to that of B) leads to the tree \mathfrak{T}' (c). The image obtained from \mathfrak{T}' (d) and its tree of shapes $\widehat{\mathfrak{T}}$ (e). (1) The tree \mathfrak{T}' (c) differs from $\widehat{\mathfrak{T}}$ (e). (2) The sign of the gradients has been modified between \mathcal{F} and $\widehat{\mathcal{F}}$ for some contours (d, in red). (Figure adapted from [17].)

5 Shifting One Node of the Tree of Shapes

We now explain how to modify the altitude of (i.e. *shift*) a node of a tree of shapes while satisfying (P_1) and (P_2) .

5.1 Notations

Let $\mathcal{F} : \mathbb{U} \rightarrow \mathbb{V}$ be an image. Let $\mathfrak{T} = (\Theta, \triangleleft)$ be its tree of shapes. Let $X \in \Theta$ be a node of \mathfrak{T} . From Property 6, we assume without loss of generality that $Alt(X) = 0$. (See Fig. 3.)

Definition 7 (Parent set). *The parent set of X is $\mathcal{P}(X) = \{P \in \Theta \mid X \triangleleft P\}$.*

If $\mathcal{P}(X) = \{P\}$, then P is the parent node of X , i.e. $X \triangleleft P$. In that case, by abuse of notation, we write $\mathcal{P}(X) = P$ and we note $m = Alt(P)$. If $\mathcal{P}(X) = \emptyset$, then X is the root of \mathfrak{T} and we set $m = 0$.

Definition 8 (Children set). *The children set of X is $\mathcal{C}(X) = \{C_i \in \Theta \mid C_i \triangleleft X\}_{i=1}^{\delta}$ (with $\delta \geq 0$).*

For any $i \in \llbracket 1, \delta \rrbracket$, we note v_i the altitude value of the node C_i , i.e. $v_i = Alt(C_i)$. We have $v_i \neq 0$, i.e. either $v_i > 0$ or $v_i < 0$.

We assume that $\{C_i\}_{i=1}^{\delta}$ is sorted with respect to the values v_i , i.e. for all $j, k \in \llbracket 1, \delta \rrbracket$, we have

$$(j < k) \implies (v_j \leq v_k) \tag{10}$$

We set $\alpha = |\{C_i \mid v_i < 0\}|$ and $\beta = |\{C_i \mid v_i > 0\}|$ (with $\delta = \alpha + \beta$). In particular, we have $\{C_i \mid v_i < 0\} = \{C_i\}_{i=1}^{\alpha}$ and $\{C_i \mid v_i > 0\} = \{C_i\}_{i=\alpha+1}^{\alpha+\beta}$. For each $i \in \llbracket 1, \alpha \rrbracket$, we set $C_i^- = C_{\alpha-i+1}$ and we note $v_i^- = v_{\alpha-i+1}$. For each

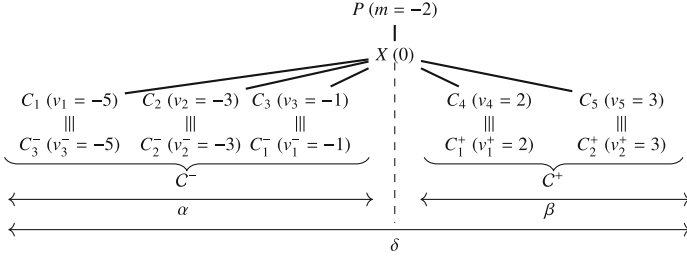


Fig. 3. Graphical example of notations of Sect. 5.1. The altitudes of nodes are given into brackets. In this figure, we have $m^+ = v_4$ and $m^- = v_3$. They correspond to the closest (higher and lower) altitudes with respect to X within the set of its children. We have $M^+ = \{C_4\}$ and $M^- = \{C_3\}$, which are the sets of children nodes associated to the respective values m^+ and m^- . Finally, we have $\mu^+ = v_4$ and $\mu^- = v_3$ which correspond to the closest (higher and lower) altitudes with respect to X within the whole set of its neighbouring nodes (parent and children).

$i \in \llbracket 1, \beta \rrbracket$, we set $C_i^+ = C_{\alpha+i}$ and we note $v_i^+ = v_{\alpha+i}$. We set $\mathcal{C}^- = \{C_i^-\}_{i=1}^\alpha$ and $\mathcal{C}^+ = \{C_i^+\}_{i=1}^\beta$. We set

$$m^+ = \min \left\{ v_i^+ \right\}_{i=1}^\beta = \begin{cases} v_1^+ & \text{if } \beta > 0 \\ +\infty & \text{if } \beta = 0 \end{cases} \quad \text{and} \quad m^- = \max \left\{ v_i^- \right\}_{i=1}^\alpha = \begin{cases} v_1^- & \text{if } \alpha > 0 \\ -\infty & \text{if } \alpha = 0 \end{cases} \quad (11)$$

and

$$\mu^+ = \begin{cases} \min\{m^+, m\} & \text{if } m > 0 \\ m^+ & \text{if } m \leq 0 \end{cases} \quad \text{and} \quad \mu^- = \begin{cases} \max\{m^-, m\} & \text{if } m < 0 \\ m^- & \text{if } m \geq 0 \end{cases} \quad (12)$$

Finally, we set

$$\mathcal{M}^+ = \{C_i^+ \in \mathcal{C}^+ \mid v_i^+ = \mu^+\} \quad \text{and} \quad \mathcal{M}^- = \{C_i^- \in \mathcal{C}^- \mid v_i^- = \mu^-\} \quad (13)$$

5.2 Short Range Shifting

Let us first suppose that we want to shift a node $X \in \Theta$ with no impact beyond its immediate vicinity. The following property states that if this shifting is small enough, then the tree of shapes remains unchanged.

Property 9. *If the new value v of $\text{Alt}(X)$ satisfies $\mu^- < v < \mu^+$, then neither \mathfrak{T} nor ρ are modified, and Alt is modified only for X .*

Now, let us suppose that we want to shift X slightly further. More precisely, we want to set the new altitude v of X so that $v = \mu^\star$ (where \star is either $+$ or $-$). Three (mutually exclusive) distinct cases can occur (Fig. 4):

- (i) $\mu^\star = m = m^\star$ (see Property 10);
- (ii) $\mu^\star = m \neq m^\star$ (see Property 11);
- (iii) $\mu^\star = m^\star \neq m$ (see Property 12).

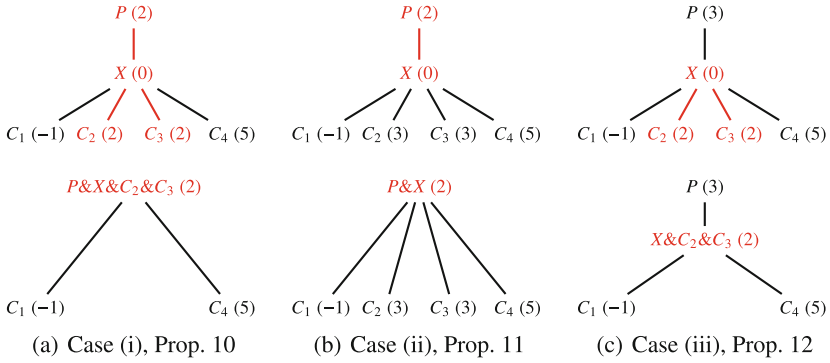


Fig. 4. The three cases of short range shifting. First row: initial configuration. Second row: final configuration. (a–c) We have $\mu^- = -1$, $\mu^+ = 2$ and the targeted altitude is $v = 2$. The part of the tree impacted by the shifting is depicted in red. (Color figure online)

In each case, it is necessary to discard nodes, as the new altitude v of X is either that of its parent (ii), of some of its children (iii), or both (i). The node discarding procedure is defined in Func. Discard. For a node $X \in \Theta$ to be discarded, it proceeds as follows: $\rho(X)$ is added to the proper part of the parent P of X (Line 1); X is removed from Θ (Line 2); (X, P) is removed from \triangleleft (Line 3); for each children C of X , (C, X) is removed from \triangleleft and replaced by (C, P) (Lines 4–6). From now on, “Discarding a node X ” will mean that we apply Func. Discard for X .

Algorithm 1 provides the general short range shifting process that corresponds to Cases (i–iii) and Properties 10–12.

Property 10. *If the new value v of X is $\mu^* = m = m^*$, then the tree of shapes of the image \mathcal{F} is modified as follows: (1) each $C \in \mathcal{M}^*$ is discarded; (2) X is discarded.*

In that case, the targeted altitude of X reaches both the altitude of some of its children and of its parent. We discard these children (Algorithm 1, Lines 4–5) and then X (Algorithm 1, Line 6).

Function Discard

Input: $\Theta, \triangleleft, \rho, X$ **Output:** $\Theta, \triangleleft, \rho$

```

1  $\rho(P) := \rho(P) \cup \rho(X)$ 
2  $\Theta \rightarrow X$ 
3  $\triangleleft \rightarrow (X, P)$ 
4 foreach  $C \triangleleft X$  do
5    $\triangleleft \rightarrow (C, X)$ 
6    $\triangleleft \leftarrow (C, P)$ 

```

Algorithm 1: Short range shifting of a node

Input: $\Theta, \triangleleft, \rho, Alt, X, v \in \llbracket \mu^-, \mu^+ \rrbracket$ **Output:** $\Theta, \triangleleft, \rho, Alt$

```

1 if  $\mu^- < v < \mu^+$  then  $Alt(X) := v$  (Property 9)
2 else
3   if  $v = m^*$  then
4     foreach  $S \in \mathcal{M}^*$  do  $Discard(\Theta, \triangleleft, \rho, S)$  (Cases (i), (iii), Properties 10, 12)
5      $Alt(X) := v$ 
6   if  $v = m$  then  $Discard(\Theta, \triangleleft, \rho, X)$  (Cases (i), (ii), Properties 10, 11)

```

Algorithm 2: Long range shifting

Input: $\Theta, \triangleleft, \rho, Alt, X, v$ **Output:** $\Theta, \triangleleft, Alt, \rho$

```

1 while  $Alt(X) \neq v$  do
2    $P := \mathcal{P}(X)$ 
3   if  $v > \mu^+$  then  $b := \mu^+$ 
4   else if  $v < \mu^-$  then  $b := \mu^-$ 
5   else  $b := v$ 
6   Apply Algorithm 1 ( $\Theta, \triangleleft, \rho, Alt, X, b$ )
7   if  $X \notin \Theta$  then  $X := P$ 

```

Property 11. *If the new value v of X is $\mu^* = m \neq m^*$, then the tree of shapes of the image \mathcal{F} is modified as follows: X is discarded.*

In that case, the targeted altitude of X reaches that of its parent only. We discard X (Algorithm 1, Line 6).

Property 12. *If the new value v of X is $\mu^* = m^* \neq m$, then the tree of shapes of the image \mathcal{F} is modified as follows: each $C \in \mathcal{M}^*$ is discarded.*

In that case, the targeted altitude of X reaches that of some of its children only. We discard these children (Algorithm 1, Lines 4–5).

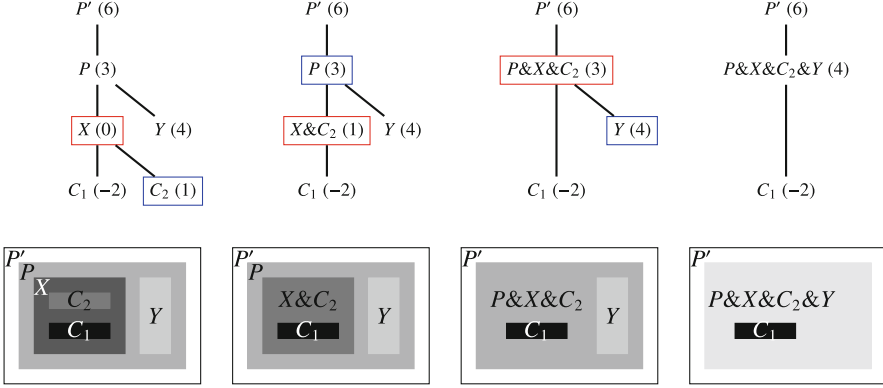


Fig. 5. The successive steps of long-range shifting for a node X (first row) and the side effect on the image (second row). The target altitude is $v = 4$. The red nodes are shifted, while the blue nodes are involved in the intermediate short-range shifting. (Color figure online)

5.3 Long Range Shifting of a Node

Let us now suppose that we want to shift a node $X \in \Theta$ by modifying its altitude $Alt(X)$ to a value $v \notin \llbracket \mu^-, \mu^+ \rrbracket$. We have either $v > \mu^+$ or $v < \mu^-$. Let us suppose that $v > \mu^+$ (the same reasoning holds for $v < \mu^-$). If we apply Algorithm 1 to the node X to modify its altitude to the value $v = \mu^+$, then the node X is merged with some of its neighbouring nodes (parent and/or children), leading to a new node \widehat{X} that differs from X but such that $X \subset \widehat{X}$. Thus, further modifying X boils down to modifying \widehat{X} . This node \widehat{X} is associated to its own interval $\llbracket \widehat{\mu}^-, \widehat{\mu}^+ \rrbracket$ with $\mu^+ < \widehat{\mu}^+$. If $v \leq \widehat{\mu}^+$, then the process ends by applying Algorithm 1 (Line 1, Property 9) with the value v . If $v > \widehat{\mu}^+$, then the process continues one step further by applying Algorithm 1 (Lines 3–6, Properties 10–12) with the value $\widehat{\mu}^+$, and so on. This iterative process is described in Algorithm 2 (see Fig. 5).

6 Shifting Many Nodes of the Tree of Shapes

In Sect. 5, we explained how to shift a single node of a tree of shapes. We now focus on the issue of shifting many nodes.

6.1 Criterion and Policy

A criterion is a Boolean function $Crit : \Theta \rightarrow \{True, False\}$ defined on the nodes of the tree of shapes (or, more generally, on any subset of \mathbb{U}). Its definition guides the selection of the nodes to be shifted. Such criteria, already considered in [13], often rely on attributes [4] that describe specific (spatial,

Algorithm 3: Generic tree of shapes modification

Input: $\Theta, \triangleleft, Alt, \rho$ **Parameters:** $Crit : \Theta \rightarrow \{True, False\}, Pol : \Theta \rightarrow \mathbb{V}$ **Output:** $\Theta, \triangleleft, Alt, \rho$

```

1 foreach  $X \in \Theta$  such that  $Crit(X) = True$  do
2    $v := Pol(X)$ 
3   Apply Algorithm 2 ( $\Theta, \triangleleft, \rho, Alt, X, v$ )

```

spectral. . .) properties of the nodes. Examples of criteria are given in Sect. 7 for the illustrative applications.

The policy of a tree-of-shapes modification framework is defined as a function $Pol : \Theta \rightarrow \mathbb{V}$. Given a criterion $Crit$, for any node $X \in \Theta$ such that $Crit(X) = True$, the function Pol is used to assign a new value v to $Alt(X)$. Many tree modification methods proposed in the literature (especially for the component trees) usually aimed at merging a node X with its parent node P . In that case, the function Pol was simply defined so that $Pol(X) = Alt(P)$. Here, the considered functions Pol are more versatile. Examples of policies are given in Sect. 7 for the illustrative applications.

6.2 General Approach for Shifting a Set of Nodes

Algorithm 3 defines a generic tree of shapes modification framework. The application of $Crit$ on the nodes of Θ defines a set $\mathcal{N} = \{X \in \Theta \mid Crit(X) = True\}$ (Line 1). Each node $X \in \mathcal{N}$ is assigned a new altitude $v \in \mathbb{V}$ using the function Pol (Line 2) and Algorithm 2 is then called in order to effectively shift X with respect to v (Line 3).

To keep this framework as generic as possible, we did not propose any sorting of \mathcal{N} nor tree-traversal strategy. Of course, such choices have a fundamental effect on the behaviour of the process, since the shifting of a node $X \in \Theta$ —and its possible side effects on the remainder of the tree—may (1) modify the part of \mathcal{N} not yet processed (e.g. by adding or removing nodes Y due to modifications of $Crit(Y)$) and (2) impact the way to shift these nodes (e.g. if $Pol(Y)$ has been impacted by the processing of X). Based on these considerations, the way to order \mathcal{N} , the choice to update (or not) the values of $Crit$ and Pol during the process and the way of processing the nodes (e.g. sequentially or simultaneously) are important hyperparameters that have to be set depending on the application and its purpose.

6.3 Preservation

Algorithm 1, 2, 3 and Func. Discard incrementally build upon each other. By definition, Func. Discard preserves (P_1) and (P_2) . From Properties 9–12, Algorithm 1 also preserves (P_1) and (P_2) . By construction this is also the case for Algorithm 2 and then for Algorithm 3.

Moreover, depending on the used function Pol , additional properties may be satisfied. For instance, by designing Pol so that for any $X \in \Theta$, $Pol(X) < Alt(X)$ (resp. $Pol(X) > Alt(X)$), one may build extensive (resp. anti-extensive) connected operators. The versatility of the proposed framework allows to tackle various image processing issues, as illustrated in the next section.

6.4 Computational Aspects

If we assume that the number of children is $\mathcal{O}(1)$ for the nodes of a tree of shapes (which is generally the case), then the time cost of *Funct. Discard* is $\mathcal{O}(1)$, the time cost of Algorithm 1 is $\mathcal{O}(1)$ and the time cost of Algorithm 2 is $\mathcal{O}(k)$ where k is the number of intermediate altitudes the current node has to go through. At each application of Algorithm 2, the number of nodes of the tree is reduced by at least k . It follows that, except the extra-cost induced by the computation of $Crit$ and Pol in Algorithm 3, its overall time cost is $\mathcal{O}(|\Theta|)$. The complexity of this generic approach described in Algorithm 3 then depends on the strategies adopted by the user for defining and updating $Crit$ and Pol . For well-chosen strategies (e.g. construction of attributes based on separable properties, no updating of the modified nodes...), the overall process may be carried out in linear time. Since the construction of the tree of shapes is itself a quasi-linear time process, this may allow the development of efficient strategies, in particular for handling large-scale images. Considering that the simplification process described in Algorithm 2 acts in only a part of the tree, composed of the subtree rooted at the processed node plus its upper branch, it is also possible to rely on parallel approaches, that may be deployed on distributed architectures.

7 Application Examples

We illustrate the relevance of the proposed approach for tree of shapes modification by providing three examples of induced connected operators acting on grey-level images. In these experiments, the trees of shapes were natively computed from *Higra* [22], and subsequently processed by our code (<https://github.com/jmendesf/ToSConOp>).

- The first one performs image quantization (Fig. 6). It reduces the dynamics of an image from 2^p to 2^k grey-levels, ($0 \leq k \leq p$). Here, $Crit$ always holds *True*. The quantized grey-levels are regularly sampled over $\llbracket 0, 2^p - 1 \rrbracket$ and Pol is defined so that the altitude of each node is shifted to the closest quantized value. This first application is a toy example, since pixel-wise quantization also satisfies (P_1) and (P_2) . It mainly aims to show the generality of our approach.
- The second one performs area opening (Fig. 7). It removes the nodes of the tree with smallest proper parts and, equivalently, the smallest details in the image. Here, $Crit(X) = True$ iff $\rho(X) < \lambda$ with $\lambda \in \mathbb{N}$. Pol is defined so that the altitude of X be shifted to m . The nodes X are processed by increasing size of their proper part.

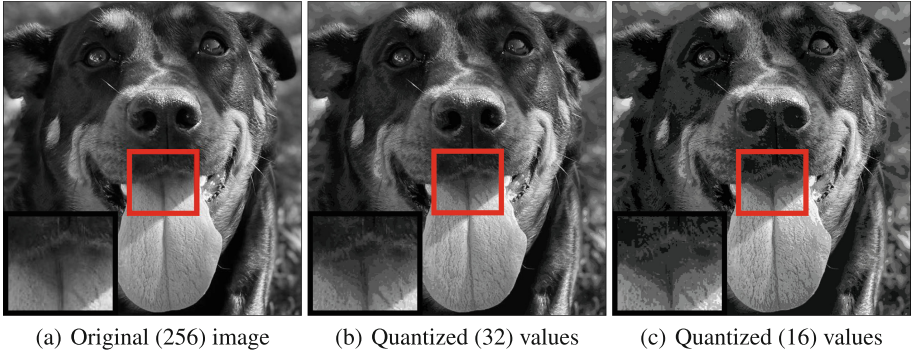


Fig. 6. (a) A natural image (256 grey levels). (b,c) The result of its quantization to 32 and 16 grey levels, respectively. (Color figure online)

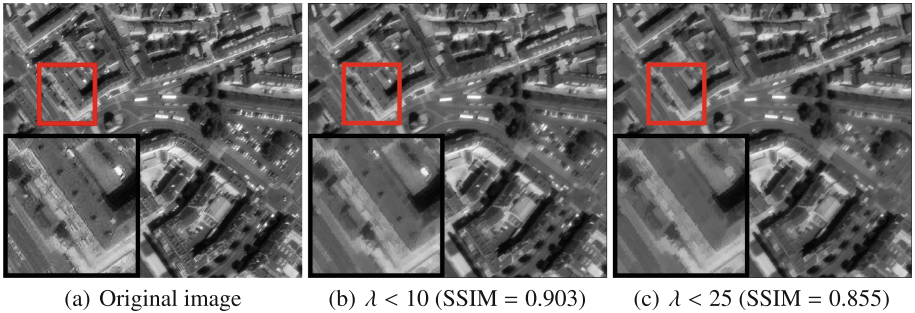


Fig. 7. (a) Remote sensing image (Landsat-7 image courtesy of the U.S. Geological Survey) ($4.1 \cdot 10^5$ pixels, tree of shapes: $1.7 \cdot 10^5$ nodes). (b,c) Simplified images obtained by area opening with (b) $\lambda = 10$ (tree of shapes: $1.5 \cdot 10^4$ nodes). (c) $\lambda = 25$ (tree of shapes: $9.7 \cdot 10^3$ nodes). The number of nodes/flat zones is progressively reduced with respect to λ in correlation with the SSIM decrease.

- The third one performs mean filtering at the scale of the nodes of the tree (Fig. 8). It progressively smooths the altitudes/grey levels between neighbouring nodes/flat zones. Here, *Crit* always holds *True*. *Pol* is defined so that X is shifted to an altitude defined as the mean value of the altitude of its neighbouring nodes. The nodes X are processed from the root to the leaves.

By definition, in each one of these applications, the designed connected operator modifies/simplifies the tree of shapes and the induced images while satisfying (P_1) and (P_2), thus leading to the preservation of the structure of the image.

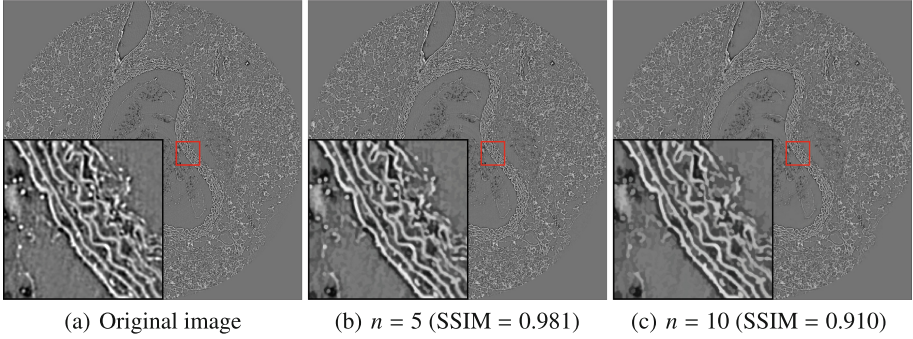


Fig. 8. (a) Biological image (synchrotron microtomography, SOLEIL ANATOMIX, project #20211303, courtesy S. Almagro) ($4.2 \cdot 10^6$ pixels, tree of shapes: $1.2 \cdot 10^6$ nodes). (b–c) Smoothed images after n iterations of mean filtering of the tree: (b) $n = 5$ (tree of shapes: $4.7 \cdot 10^5$ nodes), (c) $n = 10$ (tree of shapes: $2.3 \cdot 10^5$ nodes). The mean filter works by assigning to each node of the tree the mean value of its neighbouring nodes. The process is repeated n times. The number of nodes/flat zones is progressively reduced (removing noise) while the SSIM remains high (preserving structural information).

8 Conclusion

In this article, we introduced an approach for shifting nodes of a tree of shapes, ensuring that (P_1) the resulting tree remains the tree of shapes of the image and (P_2) the sign of the gradient of its contours is preserved. This opens the way to the development of a wide range of connected operators, designed to tackle specific issues in various applicative contexts.

The proposed framework is dedicated to the standard tree of shapes, that models grey-level images. On the one hand, the multivariate tree of shapes [7] was proposed for handling e.g. colour images. Extending our shifting paradigm to handle such multivariate images constitutes a perspective work. On the other hand, the topological tree of shapes [21] was recently introduced as a companion of the tree of shapes, which models the topology of grey-level images. Extending the proposed shifting approach to the topological tree of shapes is also a perspective work.

The ability of the proposed framework to simplify a tree of shapes, e.g. by removing (physical, semantic) noise and/or by reducing its combinatorial cost without losing significant information also opens the way to its involvement as a relevant image descriptor which could be embedded in deep-learning approaches, e.g. to model topological priors, or to ensure the preservation of structural properties of images, as already pioneered with the component tree [23].

References

1. Baderot, J., Desvignes, M., Condat, L., Dalla Mura, M.: Tree of shapes cut for material segmentation guided by a design. In: ICASSP, pp. 2593–2597 (2020)
2. Ballester, C., Caselles, V., Igual, L.: Level lines selection with variational models for segmentation and encoding. *J. Math. Imaging Vis.* **27**, 5–27 (2006)
3. Boutry, N., Najman, L., Géraud, T.: Topological properties of the first non-local digitally well-composed interpolation on n -D cubical grids. *J. Math. Imaging Vis.* **62**, 1256–1284 (2020)
4. Breen, E.J., Jones, R.: Attribute openings, thinnings, and granulometries. *Comput. Vis. Image Underst.* **64**, 377–389 (1996)
5. Carlinet, E., Crozet, S., Géraud, T.: The tree of shapes turned into a max-tree: a simple and efficient linear algorithm. In: ICIP, pp. 1488–1492 (2018)
6. Carlinet, E., Géraud, T.: A comparative review of component tree computation algorithms. *IEEE Trans. Image Process.* **23**, 3885–3895 (2014)
7. Carlinet, E., Géraud, T.: MToS: a tree of shapes for multivariate images. *IEEE Trans. Image Process.* **24**, 5330–5342 (2015)
8. Caselles, V., Monasse, P.: Grain filters. *J. Math. Imaging Vis.* **17**, 249–270 (2002)
9. Caselles, V., Meinhardt, E., Monasse, P.: Constructing the tree of shapes of an image by fusion of the trees of connected components of upper and lower level sets. *Positivity* **12**, 55–73 (2008)
10. Caselles, V., Monasse, P.: Geometric Description of Images as Topographic Maps. *Lecture Notes in Mathematics*, Springer (2010). <https://doi.org/10.1007/978-3-642-04611-7>
11. Crozet, S., Géraud, T.: A first parallel algorithm to compute the morphological tree of shapes of n D images. In: ICIP, pp. 2933–2937 (2014)
12. Géraud, T., Carlinet, E., Crozet, S., Najman, L.: A quasi-linear algorithm to compute the tree of shapes of n D. In: ISMM, pp. 98–110 (2013)
13. Jones, R.: Connected filtering and segmentation using component trees. *Comput. Vis. Image Underst.* **75**, 215–228 (1999)
14. Kurtz, C., Naegel, B., Passat, N.: Connected filtering based on multivalued component-trees. *IEEE Trans. Image Process.* **23**, 5152–5164 (2014)
15. Latecki, L.J., Eckhardt, U., Rosenfeld, A.: Well-composed sets. *Comput. Vis. Image Underst.* **61**, 70–83 (1995)
16. Monasse, P.: A root-to-leaf algorithm computing the tree of shapes of an image. In: RRPR, pp. 43–54 (2018)
17. Monasse, P., Guichard, F.: Scale-space from a level lines tree. *J. Vis. Commun. Image Represent.* **11**, 224–236 (2000)
18. Najman, L., Schmitt, M.: Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **18**, 1163–1173 (1996)
19. Pan, Y., Birdwell, J.D., Djouadi, S.M.: Preferential image segmentation using trees of shapes. *IEEE Trans. Image Process.* **18**, 854–866 (2009)
20. Passat, N., Kenmochi, Y.: A topological tree of shapes. In: DGMM, pp. 221–235 (2022)
21. Passat, N., Mendes Forte, J., Kenmochi, Y.: Morphological hierarchies: a unifying framework with new trees. *J. Math. Imaging Vis.* **65**, 718–753 (2023)
22. Perret, B., Chierchia, G., Cousty, J., Ferzoli Guimarães, S.J., Kenmochi, Y., Najman, L.: Higrá: hierarchical graph analysis. *SoftwareX* **10**, 100335 (2019)
23. Perret, B., Cousty, J.: Component tree loss function: definition and optimization. In: DGMM, pp. 248–260 (2022)

24. Rosenfeld, A.: Adjacency in digital pictures. *Inf. Control* **26**, 24–33 (1974)
25. Rosenfeld, A.: Digital topology. *Am. Math. Mon.* **86**, 621–630 (1979)
26. Salembier, P., Garrido, L.: Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE Trans. Image Process.* **9**, 561–576 (2000)
27. Salembier, P., Oliveras, A., Garrido, L.: Anti-extensive connected operators for image and sequence processing. *IEEE Trans. Image Process.* **7**, 555–570 (1998)
28. Salembier, P., Serra, J.: Flat zones filtering, connected operators, and filters by reconstruction. *IEEE Trans. Image Process.* **4**, 1153–1160 (1995)
29. Salembier, P., Wilkinson, M.H.F.: Connected operators. *IEEE Signal Process. Mag.* **26**, 136–157 (2009)
30. Song, Y., Zhang, A.: Monotonic tree. In: *DGCI*, pp. 114–123 (2002)
31. Tao, R., Qiao, J.: Fast component tree computation for images of limited levels. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**, 3059–3071 (2023)
32. Xu, Y., Géraud, T., Najman, L.: Connected filtering on tree-based shape-spaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **38**, 1126–1140 (2016)
33. Xu, Y., Géraud, T., Najman, L.: Hierarchical image simplification and segmentation based on mumford-shah-salient level line selection. *Pattern Recogn. Lett.* **83**, 278–286 (2016)



New Algorithms for Multivalued Component Trees

Nicolas Passat¹(✉), Romain Perrin², Jimmy Francky Randrianasoa^{2,3},
Camille Kurtz⁴, and Benoît Naegel²

¹ Université de Reims Champagne Ardenne, CRESTIC, Reims, France
nicolas.passat@univ-reims.fr

² Université de Strasbourg, CNRS, ICube, Strasbourg, France

³ EPITA Research Laboratory (LRE), Le Kremlin-Bicêtre, Le Kremlin-Bicêtre,
France

⁴ Université Paris Cité, LIPADE, Paris, France

Abstract. Tree-based structures can model images—and more generally valued graphs—for processing and analysis purpose. In this framework, the component tree was natively designed for grey-level images—and more generally totally ordered valued graphs. Ten years ago, the notion of a multivalued component tree was introduced to relax this grey-level/total order constraint. In this algorithmic paper, we provide new tools to handle multivalued component trees. Our contributions are twofold: (1) we propose a new algorithm for the construction of the multivalued component tree; (2) we propose two strategies for building hierarchical orders on value sets, required to further build the multivalued component trees of images/graphs relying on such value sets. Codes available at: https://github.com/bnaegel/multivalued_component_tree.

Keywords: Algorithmics · Images/valued graphs · Multivalued component trees · Hierarchical ordering · Connected operators · Mathematical morphology

1 Introduction

Building trees for modeling images is a historical research topic which was mainly investigated in field of mathematical morphology. The trees developed in this framework model in a compact way the space of the possible partitions of an image induced by its mixed spatial-spectral composition. These so-called morphological trees can be subdivided into two families, which build upon either total or partial partitions. The archetype of the first family is the binary partition tree [23] while the archetype of the second is the component tree [24].

Based on these trees, various image processing and analysis methods were developed, gathered under the name of connected operators [25, 26]. The success of morphological trees and connected operators relies on their low cost in

This work was supported by the French *Agence Nationale de la Recherche* (grants ANR-20-THIA-0006, ANR-20-CE45-0011, ANR-22-CE45-0034 and ANR-23-CE45-0015).

terms of construction and handling. Regarding their construction, they can be built in quasi-linear time [3, 23]. Regarding their involvement in image processing/analysis tasks, the two main paradigms of attribute-based node selection [2, 8] and optimal cut computation [6, 9] can be carried out in linear time.

Over the last years, efforts were geared towards enriching the framework of morphological hierarchies with new structures that generalize classical ones. In this context, the notion of multivalued component tree [10] was proposed ten years ago as a subfamily of the component graphs [16, 17] which generalize the classical component tree [24]. This new paradigm of multivalued component tree had been designed in order to build a component tree on images where values are organised with respect to a (partial) hierarchical order relation, whereas the standard component tree requires a total order.

This is an algorithmic article. It provides new tools mandatory for handling the multivalued component tree. First, we propose a new approach for building the multivalued component tree (Sect. 4). By contrast with an initial algorithm proposed in [10] (which required some pre- and post-processings to rewrite multivalued component tree construction as component tree construction), we now provide a standalone algorithm that directly builds a multivalued component tree from its multivalued image. Second, we provide two strategies for endowing a set of values with hierarchical order relations, adapted to the further construction of multivalued component trees (Sect. 5). Both strategies rely on the construction of morphological trees—component trees or binary partition trees—on/from a set of values considered itself as an image—or a valued graph.

The other parts of this article are organized as follows. In Sect. 2, we recall related works on the morphological trees. In Sect. 3, we provide the required definitions and notions related to the (multivalued) component tree. Section 6 concludes the article.

2 Related Works

A morphological tree models an image defined as a function $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ that associates to each point \mathbf{x} of its support Ω a value $\mathcal{F}(\mathbf{x})$ within the set of values \mathbb{V} . In general, Ω is endowed with an adjacency relation \sim . In other words, (Ω, \sim) is a non-directed graph. By side effect, a morphological tree can model valued graphs, and not only images.

Morphological trees are partition trees. Indeed, they are created by stacking a finite sequence of partitions of Ω . Each partition is composed of subsets $X \subseteq \Omega$ that constitute the nodes (root, internal nodes, leaves) of the tree, and a tree models the inclusion relation between them. Such trees can be classified in two main families: those originated either from (1) total partitions or (2) partial partitions of Ω .

The archetype of the total partition trees is the binary partition tree [23] (also declined under variants: α -tree [27], watershed tree [12], etc.). Except the leaves, each node is a connected subset $X \subseteq \Omega$ which has two children nodes X_1 and X_2 that form a partition of X , leading to a top-down binary decomposition

of the root Ω of the tree into subsets of decreasing size. The construction of such trees is guided by one or many [20] criteria which determine the merging order of the smallest subsets provided by an initial partition of Ω , that defines the leaves of the tree.

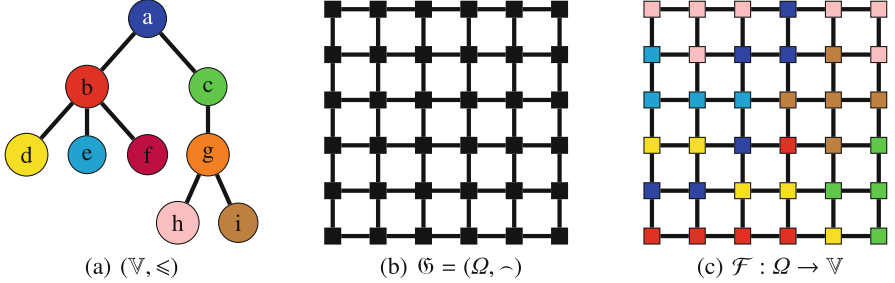


Fig. 1. (a) A set of values $\mathbb{V} = \{a, b, c, d, e, f, g, h, i\}$, endowed with a hierarchical order \leq such that the minimum is a and the maximal elements are d, e, f, h, i . This ordered set is depicted here as its Hasse diagram, which is—by definition—a tree. (b) A set $\Omega = \llbracket 0, 5 \rrbracket^2$ (squares) endowed with an adjacency \sim (segments), leading to the graph $\mathfrak{G} = (\Omega, \sim)$. (c) A multivalued image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ built on the support Ω (b) and taking its values in \mathbb{V} (a). The colour of each square is associated to the value of the corresponding point.

The archetype of the partial partition trees is the component tree [24] (also declined under variants: hyperconnection tree [18], tree of shapes [11], topological tree of shapes [14], complete tree of shapes [15], etc.). The component tree is built from successive threshold sets, at each value $v \in \mathbb{V}$ of the image \mathcal{F} . The component tree models the inclusion relation between the connected components of these threshold sets. Each node is a subset $X \subseteq \Omega$ corresponding to a connected component at a given value $v \in \mathbb{V}$. If X is not a flat zone of the image, it has $k \geq 1$ children nodes X_i ($1 \leq i \leq k$) that form a partition of a strict subset $Y \subset X$, which corresponds to the part of X where the values of \mathcal{F} are strictly greater than v .

Many efforts were dedicated to the efficient construction of morphological trees, and especially the component tree. An overview of the classical algorithms, based e.g. on flooding or union-find paradigms, can be found in [3]. A recent trend is also to develop parallel algorithms, based on distributed paradigms [4, 5] or GPU-based approaches [1].

3 Multivalued Component Tree

Let Ω be a finite set and \sim an adjacency (irreflexive, symmetric) relation on Ω that induces the (equivalence) connectedness relation by reflexive-transitive closure of \sim . The couple $\mathfrak{G} = (\Omega, \sim)$ is a non-directed graph. For any subset

$X \subseteq \Omega$, we note $\mathcal{C}[X]$ the set of the connected components (i.e. the maximal connected sets) of the subgraph (X, \curvearrowright) of \mathfrak{G} induced by X . We assume that \mathfrak{G} is connected, i.e. $\mathcal{C}[\Omega] = \{\Omega\}$. See Fig. 1(b).

Let \mathbb{V} be a finite set and \leq a hierarchical order on \mathbb{V} , i.e. an order (1) which admits a minimum (resp. a maximum) and (2) such that for any $v \in \mathbb{V}$, the subset of the elements lower (resp. greater) than v is totally ordered by \leq . See Fig. 1(a).

A total order is a hierarchical order. Thus, all the definitions given below for the multivalued component tree [10] generalize those of the classical component tree [24].

Let us consider an image \mathcal{F} defined as a function $\mathcal{F} : \Omega \rightarrow \mathbb{V}$. See Fig. 1(c). The threshold set of \mathcal{F} at value $v \in \mathbb{V}$ is defined by

$$\Lambda_v(\mathcal{F}) = \{\mathbf{x} \in \Omega \mid v \leq \mathcal{F}(\mathbf{x})\} \quad (1)$$

See Fig. 2(a). We set

$$\Theta = \bigcup_{v \in \mathbb{V}} \mathcal{C}[\Lambda_v(\mathcal{F})] \quad (2)$$

which gathers the connected components at each threshold set $\Lambda_v(\mathcal{F})$ ($v \in \mathbb{V}$). The elements of Θ are called the nodes of the multivalued component tree.

A node may be generated at many threshold values (see the ‘‘T-shaped’’ connected component in Fig. 2(a)). In particular, for any $X \in \Theta$, we set

$$\mathbb{I}(X) = \{v \in \mathbb{V} \mid X \in \mathcal{C}[\Lambda_v(\mathcal{F})]\} \quad (3)$$

and we define the remanence $\tau(X)$ of X as the number of threshold sets to which X belongs, i.e. as

$$\tau(X) = |\mathbb{I}(X)| \quad (4)$$

We also set $\omega(X)$ as the maximal value of threshold sets to which X belongs, i.e. as

$$\omega(X) = \bigvee_{\leq} \mathbb{I}(X) \quad (5)$$

For instance, for the ‘‘T-shaped’’ connected component X in Fig. 2, which belongs to $\Lambda_c(\mathcal{F})$, $\Lambda_g(\mathcal{F})$ and $\Lambda_h(\mathcal{F})$, we have $\mathbb{I}(X) = \{c, g, h\}$, the remanence of X is $\tau(X) = 3$ and we have $\omega(X) = h$, since $c \leq g \leq h$.

The inclusion relation \subseteq is a hierarchical order on Θ . We note \triangleleft the reflexive-transitive reduction of \subseteq with respect to Θ . The couple $\mathfrak{T} = (\Theta, \triangleleft)$, i.e. the Hasse diagram of (Θ, \subseteq) , is a tree called the multivalued component tree. See Fig. 2(b).

For any node $X \in \Theta$, we define the proper part $\rho(X) \subseteq \Omega$ of X as

$$\rho(X) = X \setminus \bigcup_{Y \triangleleft X} Y = \{\mathbf{x} \in \Omega \mid \mathcal{F}(\mathbf{x}) = \omega(X)\} \quad (6)$$

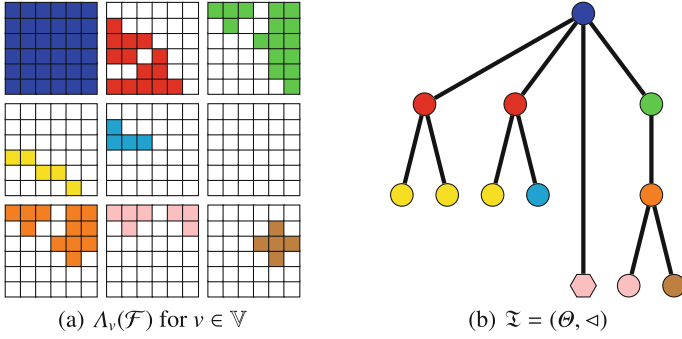


Fig. 2. (a) The nine threshold sets $\Lambda_v(\mathcal{F})$ for the image \mathcal{F} of Fig. 1(c). The squares depicted in color (resp. white) belong (resp. do not belong) to $\Lambda_v(\mathcal{F})$. Note that $\Lambda_f(\mathcal{F}) = \emptyset$ since the image \mathcal{F} has no point of value f . Also note that $\Lambda_g(\mathcal{F}) \neq \emptyset$ whereas \mathcal{F} has no point of value g . This is justified by the fact that $\Lambda_g(\mathcal{F})$ is partitioned by $\Lambda_h(\mathcal{F})$ and $\Lambda_i(\mathcal{F})$. A connected component (“T-shaped”, in the upper-left part of the image) is common to the threshold sets $\Lambda_c(\mathcal{F})$, $\Lambda_g(\mathcal{F})$ and $\Lambda_h(\mathcal{F})$. (b) The multivalued component tree $\mathfrak{T} = (\Theta, \triangleleft)$ of the image \mathcal{F} of Fig. 1(c). Each disk/hexagon corresponds to a node $X \subseteq \Omega$ of Θ (the unique hexagonal node X corresponds to the three occurrences of the “T-shaped” connected component). The color of the disk/hexagon corresponds to the value $\omega(X)$ of the node.

The multivalued component tree \mathfrak{T} is an image model of the image \mathcal{F} . Indeed, we can reconstruct \mathcal{F} from \mathfrak{T} as follows

$$\forall \mathbf{x} \in \Omega, \mathcal{F}(\mathbf{x}) = \bigwedge_{X \in \Theta}^{\leq} \mathbf{1}_{(X, \omega(X))}(\mathbf{x}) \quad (7)$$

where $\mathbf{1}_{(A, u)} : \Omega \rightarrow \mathbb{V}$ is the cylinder function defined by $\mathbf{1}_{(A, u)}(\mathbf{x}) = u$ if $\mathbf{x} \in A \subseteq \Omega$ and $\bigwedge^{\leq} \mathbb{V}$ (the minimum of (\mathbb{V}, \leq)) otherwise.

These definitions given for the multivalued component tree are similar to those of the standard component tree. The only differences are the following:

- \leq is a hierarchical order whereas it is a total order for the component tree;
- it may happen that $\rho(X) = \emptyset$ whereas we have $\rho(X) \neq \emptyset$ for the component tree.

4 Building the Multivalued Component Tree

4.1 Some Reminders of the Previous Algorithm

In [10], a first strategy had been proposed for building the multivalued component tree. The main idea was to rewrite the image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ as an image $\widehat{\mathcal{F}} : \widehat{\Omega} \rightarrow \widehat{\mathbb{V}}$ where $\widehat{\Omega} \supseteq \Omega$ and $\widehat{\mathbb{V}} = \llbracket 0, p \rrbracket \subset \mathbb{N}$ with $p \leq |\mathbb{V}|$. The set $\widehat{\mathbb{V}}$ was endowed with the total order \leq on \mathbb{N} such that there is a homomorphism from

Algorithm 1: Building the multivalued component tree

Input: $(\Omega, \curvearrowright), (\mathbb{V}, \leq), \mathcal{F} : \Omega \rightarrow \mathbb{V}$ **Output:** $\mathfrak{T} = (\Theta, \triangleleft)$

```

1 Build nodes
2 Build points
3 Build status
4 Build nb_nodes
5 Build index
6 Build progress
7  $v_{\min} := \bigwedge^{\leq} \mathbb{V}$ 
8 Choose  $\mathbf{x}_{\min} \in \Omega$  such that  $\mathcal{F}(\mathbf{x}_{\min}) = v_{\min}$ 
9 points[ $v_{\min}$ ].add( $\mathbf{x}_{\min}$ )
10 progress[ $v_{\min}$ ] := true
11 Flood( $v_{\min}$ )

```

(\mathbb{V}, \leq) to $(\widehat{\mathbb{V}}, \leq)$ induced by the equivalence relation on \mathbb{V} that gathers the values of equal distance with respect to the minimum $\bigwedge^{\leq} \mathbb{V}$ in the Hasse diagram of (\mathbb{V}, \leq) . The set $\widehat{\Omega}$ was endowed with an adjacency $\curvearrowright_{\widehat{\Omega}}$ such that there is an increasing homeomorphism from the graph $(\Omega, \curvearrowright)$ to the graph $(\widehat{\Omega}, \curvearrowright_{\widehat{\Omega}})$. The latter can be defined by adding a new vertex $\varepsilon_{\{x,y\}}$ in $\widehat{\Omega}$, and replacing the adjacency link $x \curvearrowright y$ by the two links $x \curvearrowright \varepsilon_{\{x,y\}}$ and $\varepsilon_{\{x,y\}} \curvearrowright y$, whenever the two vertices $x, y \in \Omega$ are such that $x \curvearrowright y$ while $\mathcal{F}(x)$ and $\mathcal{F}(y)$ are non-comparable with respect to \leq . It was proved that the component tree $\widehat{\mathfrak{T}}$ of $\widehat{\mathcal{F}}$ is isomorphic with the multivalued component tree \mathfrak{T} of \mathcal{F} . It was then possible to build a multivalued component tree by using any algorithm dedicated to the construction of the component tree, at the cost of (1) the preprocessing that builds $\widehat{\mathcal{F}}$ from \mathcal{F} , and (2) a post-processing that retrieves $\mathfrak{T} = (\Theta, \triangleleft)$ from $\widehat{\mathfrak{T}} = (\widehat{\Theta}, \widehat{\triangleleft})$ by removing from the proper part $\rho(X)$ of each node $X \in \widehat{\Theta}$ the elements of $X \setminus \Omega$ and by substituting the values of \mathbb{V} to those of $\widehat{\mathbb{V}}$ in the definition of $\omega(X)$.

4.2 A New Algorithm

We now present a new alternative algorithm that no longer requires such preconditioning of the image \mathcal{F} . The construction scheme is detailed in Alg. 1 and Func. `Flood`. The proposed strategy is derived from the component tree construction presented by Salembier et al. in [24]. It also finds inspiration in the mask-based algorithm developed by Ouzounis et al. in [13].

The proposed algorithm relies on the following data structures:

- **nodes**: a 2D array which stores the nodes of the multivalued component tree. The first dimension is indexed by the values of \mathbb{V} . The second dimension is indexed by the identifiers of the nodes. In other words, **nodes** encodes Θ ; **nodes**[v] encodes the nodes of Θ at value v ; and **nodes**[v][i] encodes the i th node of Θ at value v ;

Function Flood

```

Input:  $u \in \mathbb{V}$ : current level
Output:  $w \in \mathbb{V}$ : value of the parent node of the root of the built (partial)
multivalued component tree at value  $u$  (or  $\varepsilon$  if the node has no parent)
1 while  $!(\text{points}[u].\text{empty}())$  do
2    $\mathbf{x} := \text{points}[u].\text{remove}()$ 
3   if  $\text{index}[u] > \text{nb\_nodes}[u]$  then
4      $\text{nb\_nodes}[u] := \text{index}[u]$  // in practice,  $\text{nb\_nodes}[u]++$ 
5      $X := \text{create\_node}()$  // new node in  $\Theta$ 
6      $\text{nodes}[u].\text{insert}(X)$ 
7   if  $\mathcal{F}(\mathbf{x}) \neq u$  then
8      $w := \mathcal{F}(\mathbf{x})$ 
9      $\text{points}[w].\text{add}(\mathbf{x})$ 
10     $\text{progress}[w] := \text{true}$ 
11    while  $u < w$  do  $w := \text{Flood}(w)$ 
12  else
13     $\text{status}[\mathbf{x}] := \text{index}[u]$ 
14     $\text{nodes}[u][\text{index}[u]].\text{add\_to\_proper\_part}(\mathbf{x})$ 
15    foreach  $\mathbf{y} \in \mathbf{x}$  do
16       $w := \mathcal{F}(\mathbf{y})$ 
17      if  $\text{status}[\mathbf{y}] = -1$  then
18        if  $u \leq w$  then  $\hat{w} := w$ 
19        else  $\hat{w} := \bigwedge^{\leq} \{u, w\}$ 
20         $\text{points}[\hat{w}].\text{add}(\mathbf{y})$ 
21         $\text{status}[\mathbf{y}] := 0$ 
22         $\text{progress}[\hat{w}] := \text{true}$ 
23        while  $u < \hat{w}$  do  $\hat{w} := \text{Flood}(\hat{w})$ 
24  if  $u = v_{\min}$  then
25     $w := \varepsilon$ 
26  else
27     $w := \bigvee^{\leq} \{w' \in \mathbb{V} \mid w' < u\}$ 
28    while  $\text{progress}[w] = \text{false}$  do  $w := \bigvee^{\leq} \{w' \in \mathbb{V} \mid w' < w\}$ 
29     $\text{create\_edge}(\text{nodes}[u][\text{index}[u]], \text{nodes}[w][\text{index}[w]])$  // new edge in
     $\triangleleft$ 
30   $\text{progress}[u] = \text{false}$ 
31   $\text{index}[u]++$ 
32  return  $w$ 

```

- **points**: a 2D array which stores the processed points of the image. The first dimension is indexed by the values of \mathbb{V} . In other words, $\text{points}[v]$ encodes all the points $\mathbf{x} \in \Omega$ currently processed “at value v ”;
- **status**: a 1D array which stores the status of each point of the image. For any point $\mathbf{x} \in \Omega$, we have $\text{status}[\mathbf{x}] = -1$ if \mathbf{x} is unprocessed; $\text{status}[\mathbf{x}]$

- 0 if \mathbf{x} belongs to `points`; and `status`[\mathbf{x}] = $i > 0$ if \mathbf{x} belongs to the proper part $\rho(X)$ of the node X stored in `nodes`[$\mathcal{F}(\mathbf{x})$][i];
- `nb_nodes` and `index`: two 1D arrays which store the number of nodes already fully built and the index of the node currently built at each value of \mathbb{V} , respectively;
- `progress`: a 1D array which indicates if there exists a node at value v , currently under construction or to be built, which is an ancestor of the node at value u currently being defined.

By comparison with the component tree construction detailed in [24], the one proposed here for multivalued component tree construction differs with regard to `Flood` as follows:

- In [24], we have $\mathbf{x} \in \text{nodes}[\mathcal{F}(\mathbf{x})]$. Here (Lines 7–11), we may have $\mathbf{x} \in \text{nodes}[u]$ with $u \neq \mathcal{F}(\mathbf{x})$. This happens when \mathbf{x} is stored in `nodes` as the neighbour of another point with a non-comparable value. In such case, the chosen value u is the infimum of these two non-comparable values, and we have in particular $u < \mathcal{F}(\mathbf{x})$;
- For two adjacent points $\mathbf{x} \frown \mathbf{y}$, it may occur that $\mathcal{F}(\mathbf{x})$ and $\mathcal{F}(\mathbf{y})$ be non-comparable. In particular, the “else” case at Line 19 means that either $u > w$ or u and w are non-comparable. In the first case, \hat{w} is set to w . In the second case, \hat{w} is the infimum of u and w , distinct from them. In this last case, the point \mathbf{y} of value w is added to `nodes`[\hat{w}] and not to `nodes`[w]. This will further result in the scenario discussed above (Lines 7–11).

Note that `nodes`, `points`, `status`, `nb_nodes`, `index`, `progress`, are handled as global variables. In practice, `Flood` is then called for an input value v , *with a given configuration of these variables* and modifies them.

An example of the behaviour of the algorithm is provided in Figs. 3 and 4. For the image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ of Fig. 3(a), the processing order of the points of Ω is given in Fig. 3(b), from the first (1) to the last one (36). Figure 4 shows the progress of the construction of the multivalued component tree of \mathcal{F} with respect to the processed points.

4.3 Complexity Analysis

In this analysis, we assume that $|\frown| = \mathcal{O}(|\Omega|)$, which is the case in digital images. We note $\kappa(\mathbb{V})$ the time cost required to compare two elements of \mathbb{V} or to compute their infimum. Depending on the way (\mathbb{V}, \leq) is modeled, $\kappa(\mathbb{V})$ may vary from $\mathcal{O}(1)$ (with a space cost of $\mathcal{O}(|\mathbb{V}|^2)$) to $\mathcal{O}(\log |\mathbb{V}|)$ or $\mathcal{O}(|\mathbb{V}|)$ (depending on the equilibrium of the Hasse diagram, with a space cost of $\mathcal{O}(|\mathbb{V}|)$). We note $h(\mathbb{V}) \in \mathbb{N}$ the height of the Hasse diagram of (\mathbb{V}, \leq) .

Regarding the data structures:

- The size of `nodes` is $\mathcal{O}(|\Omega|)$. It is initialized with a time cost $\mathcal{O}(1)$. When accessing `nodes`[v] for reading or writing, the induced time cost is $\mathcal{O}(\log(|\Omega|))$.

- The size of **points** is $\mathcal{O}(|\Omega|)$. It is initialized with a time cost $\mathcal{O}(|\Omega| \cdot \kappa(|\mathbb{V}|))$. When accessing a set **points**[v] for reading or writing, the induced time cost is $\mathcal{O}(1)$.
- The size of **status** is $\mathcal{O}(|\Omega|)$. It is initialized with a time cost $\mathcal{O}(|\Omega|)$. Accessing it for reading or writing has a time cost $\mathcal{O}(1)$.
- The size of **nb_nodes** and **index** is $\mathcal{O}(|\Omega|)$. They are initialized with a time cost $\mathcal{O}(1)$. Accessing them for reading or writing has a time cost $\mathcal{O}(\log(|\Omega|))$.
- The size of **progress** is $\mathcal{O}(h(\mathbb{V}))$. It is initialized with a time cost $\mathcal{O}(1)$. Accessing it for reading or writing has a time cost $\mathcal{O}(\log(h(\mathbb{V})))$.

Based on these considerations, the time cost for Alg. 1 (except Line 11) is $\mathcal{O}(|\Omega| \cdot \kappa(|\mathbb{V}|))$.

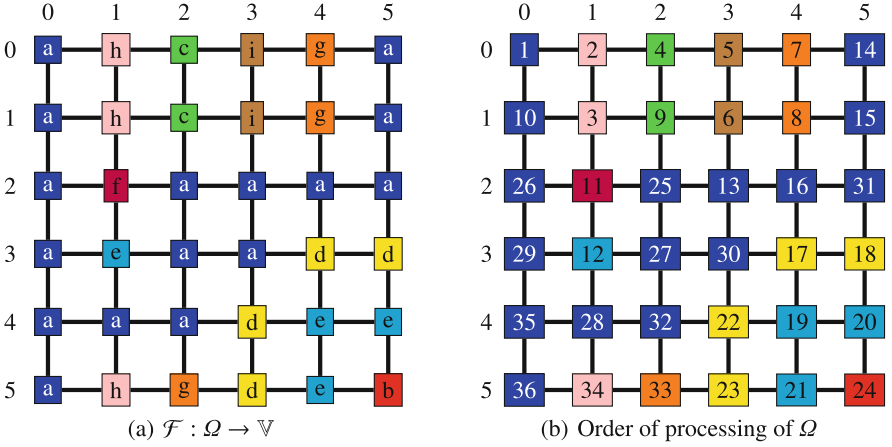


Fig. 3. (a) An image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$, following the same conventions as in Fig. 1. (b) The order of processing of the points of Ω by Alg. 1, from the first processed point (“1”) to the last processed point (“36”). At Line 15 of Alg. 1, the points \mathbf{y} adjacent to \mathbf{x} are considered in the clockwise order, starting from the point on the right of \mathbf{x} .

The time cost of **Flood** depends on:

- the size of Θ . In particular, for each node of Θ , **Flood** is called once, with an induced time cost $\mathcal{O}(\log(|\Omega|) + \log(h(\mathbb{V})))$ related to Lines 1 and 30–31;
- the size of \triangleleft . In particular, for each edge of \triangleleft , **Flood** is called once, with an induced time cost $\mathcal{O}(\tau(X) \cdot (\log(h(\mathbb{V}))) + \kappa(|\mathbb{V}|))$ related to Lines 24–29 (where X is the node associated to the processed edge (X, Y));
- the number of points of Ω . In particular, for each point $\mathbf{x} \in \Omega$, the while loop of **Flood** (Lines 2–23) is run once or twice, with an induced time cost $\mathcal{O}(\log(|\Omega|) + \log(h(\mathbb{V})) + \kappa(\mathbb{V}))$.

It follows that the overall time cost of the construction process (Alg. 1 and Func. Flood) is

$$\mathcal{T} = \mathcal{O}\left(|\Omega| \cdot \left(\log(|\Omega|) + h(\mathbb{V}) \cdot \log(h(\mathbb{V})) + h(\mathbb{V}) \cdot \kappa(|\mathbb{V}|)\right)\right) \quad (8)$$

If the Hasse diagram of (\mathbb{V}, \leq) is well balanced, the time cost becomes

$$\mathcal{T} = \mathcal{O}\left(|\Omega| \cdot \left(\log(|\Omega|) + (\log(|\mathbb{V}|))^2\right)\right) \quad (9)$$

The algorithms dedicated to build the standard component tree present a quasi-linear computational cost $\mathcal{O}(|\Omega| \cdot \log(|\Omega|))$. The initial algorithm dedicated to build the multivalued component tree [10] (see Sect. 4.1) relies on such quasi-linear time cost algorithms. In addition, it requires a pre- and a post-processing step. During the pre-processing, the support of the image is extended from Ω to $\widehat{\Omega}$, and the time cost of the subsequent component tree construction is then $\mathcal{O}(|\widehat{\Omega}| \cdot \log(|\widehat{\Omega}|))$. We have $|\widehat{\Omega}| \geq |\Omega|$, and the size of $\widehat{\Omega}$ depends on the size of the subset of edges of \wedge that link vertices of Ω with non-comparable values. More precisely, we have $|\widehat{\Omega}| = |\Omega|$ in the best scenario, i.e. when all the couples of adjacent vertices $\mathbf{x} \wedge \mathbf{y}$ are such that $\mathcal{F}(\mathbf{x})$ and $\mathcal{F}(\mathbf{y})$ are comparable. By contrast, we have $|\widehat{\Omega}| = |\Omega| + \mathcal{O}(|\wedge|)$ when all the couples of adjacent vertices $\mathbf{x} \wedge \mathbf{y}$ are such that $\mathcal{F}(\mathbf{x})$ and $\mathcal{F}(\mathbf{y})$ are non-comparable. This last case may generally occur whenever the Hasse diagram of (\mathbb{V}, \leq) is well-balanced. In the case of a d -dimensional digital image, the size of \wedge is $d \cdot |\Omega|$. In this context, the overall time cost of the computation of the multivalued component-tree is $\mathcal{O}(d \cdot |\Omega| \log |\Omega|)$. We observe in particular that the initial algorithm [10] and the new one proposed here are not sensitive to the same parameters. The first has a time cost that progressively degrades while the dimension of the image increases, while the second has a time cost that progressively degrades while the size of the value space increases. It follows that both algorithms are complementary, and may be considered depending on the application hypotheses.

5 Hierarchical Order Construction

Building the multivalued component tree of an image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ requires a hierarchical order on the set of values \mathbb{V} . In this section, we discuss the ways to endow \mathbb{V} with such hierarchical orders (or, more generally, preorders) \leq . Two strategies are proposed:

- building a preorder \leq on the only values of \mathbb{V} (Sect. 5.1);
- enriching \mathbb{V} with additional values leading to a larger set \mathbb{W} and defining an order \leq on \mathbb{W} so that the values of \mathbb{V} are the maximal elements of (\mathbb{W}, \leq) (Sect. 5.2).

5.1 (Pre)ordering the Value Set

We first aim to build a hierarchical preorder $\leq_{\mathbb{V}}$ on \mathbb{V} . Equivalently, we must set:

- an equivalence relation \sim on \mathbb{V} that gathers values which are mutually and symmetrically comparable, leading to a quotient set \mathbb{V}/\sim , noted \mathbb{K} ;
- a hierarchical order $\leq_{\mathbb{K}}$ on \mathbb{K} .

This preorder $\leq_{\mathbb{V}}$ is then defined, for all $u, v \in \mathbb{V}$, by

$$(u \leq_{\mathbb{V}} v) \iff ([u]_{\sim} \leq_{\mathbb{K}} [v]_{\sim}) \quad (10)$$

Let us come back to the notion of a component tree (see Sect. 3 by assuming that \leq is a total order). We consider a graph $\mathfrak{G}_{\Delta} = (\Delta, \wedge_{\Delta})$ where Δ is a finite set and \wedge_{Δ} is an adjacency on Δ , and a function $\delta : \Delta \rightarrow \mathbb{N}$ (with \mathbb{N} endowed with the usual \leq relation). Following Sect. 3, one can build the component tree $\mathfrak{T}_{\Delta} = (\Theta_{\Delta}, \triangleleft_{\Delta})$ of $(\mathfrak{G}_{\Delta}, \delta)$.

The set Θ_{Δ} is a cover of Δ . More precisely, we have $\bigcup \Theta_{\Delta} = \Delta$ and $\forall A \in \Theta_{\Delta}, A \neq \emptyset$. However, two distinct elements $A, B \in \Theta_{\Delta}$ may have a non-empty intersection. Indeed, for all $A, B \in \Theta_{\Delta}$ we have $A \cap B \neq \emptyset \Rightarrow A \subseteq B \vee B \subseteq A$. This last point may prohibit Θ_{Δ} to be a partition of Δ . Nonetheless, we can define the set Δ^* from Δ composed of the (non-empty) proper parts of the nodes of Θ_{Δ} . Given a node $A \in \Theta_{\Delta}$, the subset $A^* = \rho(A) \subseteq A$ is defined as in Eq. (6). The set Θ_{Δ}^* is then defined as

$$\Theta_{\Delta}^* = \{A^* \mid A \in \Theta\} \quad (11)$$

In particular, the application that maps Θ_{Δ} onto Θ_{Δ}^* is a bijection, and Θ_{Δ}^* is a partition of Δ . It follows that Θ_{Δ} defines an equivalence relation \sim_{Δ} on Δ .

The component tree $(\Theta_{\Delta}, \triangleleft_{\Delta})$ is the Hasse diagram of the ordered set $(\Theta_{\Delta}, \subseteq)$. Since Θ_{Δ} and Θ_{Δ}^* are in bijection, we can derive the order \subseteq^* on Θ_{Δ}^* by

$$(A^* \subseteq^* B^*) \iff (A \subseteq B) \quad (12)$$

The Hasse diagram $(\Theta_{\Delta}^*, \triangleleft_{\Delta}^*)$ of $(\Theta_{\Delta}^*, \subseteq^*)$ is then isomorphic to the Hasse diagram $(\Theta_{\Delta}, \triangleleft_{\Delta})$, i.e. the component tree of $(\mathfrak{G}_{\Delta}, \delta)$.

Following the notations given at the beginning of this section, and setting $\Delta = \mathbb{V}$, $\sim_{\Delta} = \sim$, $\mathbb{K} = \Theta_{\Delta}^*$ and $\subseteq^* = \leq_{\mathbb{K}}$, we can define a hierarchical preorder $\leq_{\mathbb{V}}$ on \mathbb{V} from a component tree. In particular, it is only required that \mathbb{V} be endowed with the two elements necessary for building this component tree, namely:

- an adjacency relation $\wedge_{\mathbb{V}}$, allowing to map a graph structure on \mathbb{V} ;
- a function $\delta_{\mathbb{V}} : \mathbb{V} \rightarrow \mathbb{N}$, allowing to associate to each element of \mathbb{V} a value within the totally ordered set (\mathbb{N}, \leq) .

Example. Let us consider a colour image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ where the colour values are encoded in the 8-bit per band RGB space $\mathbb{V} = \llbracket 0, 255 \rrbracket^3$. We model \mathbb{V} as the RGB cube, where each colour $v = (r, g, b) \in \llbracket 0, 255 \rrbracket^3$ corresponds to a point in the Cartesian space. We endow \mathbb{V} with the standard 6-adjacency $\wedge_{\mathbb{V}}$ defined in digital topology [22], that models the 1-distance between two colours with respect to the ℓ_1 norm. We set $\mathfrak{G}_{\Delta} = (\mathbb{V}, \wedge_{\mathbb{V}})$. Let us define $\delta_{\mathbb{V}}$ as the histogram of the image \mathcal{F} . In other words, for any colour $v \in \mathbb{V}$, we set $\delta_{\mathbb{V}}(v) = |\{\mathbf{x} \in \Omega \mid \mathcal{F}(\mathbf{x}) = v\}|$.

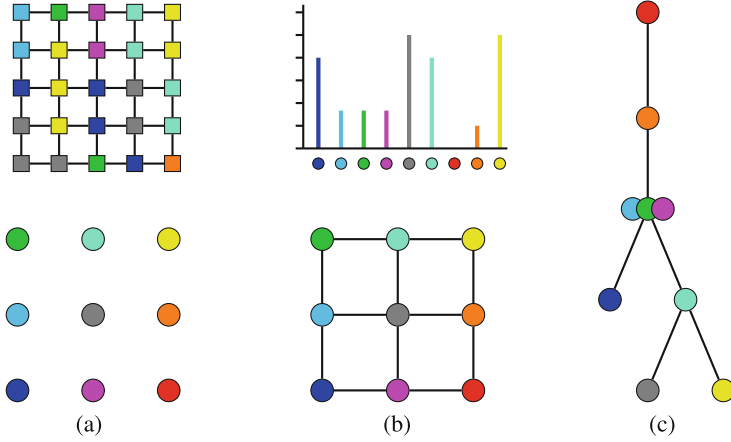


Fig. 5. Illustration of the construction of a hierarchical preorder on a value set (see Sect. 5.1). (a) Top: an image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$. The set Ω is equal to $\llbracket 0, 4 \rrbracket^2$ and is endowed with an adjacency relation \sim corresponding to the 4-adjacency. Bottom: the set \mathbb{V} composed of 9 values. (b) Top: the histogram of the image \mathcal{F} , used as function $\delta_{\mathbb{V}} : \mathbb{V} \rightarrow \mathbb{N}$. Bottom: the set \mathbb{V} is endowed with an adjacency $\sim_{\mathbb{V}}$. (c) The component tree of $(\mathbb{V}, \delta_{\mathbb{V}})$ seen as an image from the set of values \mathbb{V} to \mathbb{N} where \mathbb{V} is endowed with $\sim_{\mathbb{V}}$. This component tree defines a hierarchical preorder \leq , where the red value is the minimum, the dark blue, yellow and grey values are the maximal elements, and where the three values light blue, green and fushia are mutually greater and lower. Once \mathbb{V} is endowed with \leq , it becomes possible to compute the multivalued component tree of the image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$ of (a) with respect to \leq .

Based on the above discussion, the component tree $\mathfrak{T}_{\Delta} = (\Theta_{\Delta}, \triangleleft_{\Delta})$ built from $(\mathfrak{G}_{\Delta}, \delta_{\mathbb{V}})$ defines the Hasse diagram of a hierarchical preorder $\leq_{\mathbb{V}}$ on \mathbb{V} . This example is illustrated in a simplified version in Fig. 5. From \mathcal{F} and $\leq_{\mathbb{V}}$, it is then possible to build the multivalued component tree of \mathcal{F} induced by its histogram.

5.2 Ordering the Enriched Value Set

We now aim to build a hierarchical (pre)order $\leq_{\mathbb{W}}$ on a superset \mathbb{W} of \mathbb{V} so that $\mathbb{V} = \nabla^{\leq_{\mathbb{W}}} \mathbb{W}$, i.e. the elements of \mathbb{V} are the maximal elements with respect to $\leq_{\mathbb{W}}$.

The smallest set \mathbb{W} that can be proposed is $\mathbb{W} = \mathbb{V} \cup \{\perp\}$ where $\perp \notin \mathbb{V}$ is a unique element added to \mathbb{V} that acts as minimum for $\leq_{\mathbb{W}}$ (such paradigm was investigated in [21]). The induced hierarchical preorder (which is indeed an order) is the relation $\leq_{\mathbb{W}}$ defined as $\{(\perp, v) \mid v \in \mathbb{V}\}$ with $\perp = \bigwedge^{\leq_{\mathbb{W}}} \mathbb{W}$ and $\mathbb{V} = \nabla^{\leq_{\mathbb{W}}} \mathbb{W}$. It is possible to build larger supersets \mathbb{W} and to endow them with hierarchical preorders $\leq_{\mathbb{W}}$ that also fulfill the above assumption. Such sets \mathbb{W} can be of arbitrary size.

We first observe that defining a preorder instead of an order is not relevant (by contrast with Sect. 5.1). Let \mathbb{W} be a set and $\leq_{\mathbb{W}}$ a hierarchical preorder on

\mathbb{W} . We assume that $\nabla^{\leq_{\mathbb{W}}} \mathbb{W} = \mathbb{V}$ and $\bigwedge^{\leq_{\mathbb{W}}} \mathbb{W} = \perp$, with $\perp \in \mathbb{W} \setminus \mathbb{V}$. For any $w \in \mathbb{W}$, we set $\mathbb{V}_w = \{v \in \mathbb{V} \mid w \leq_{\mathbb{W}} v\}$ (note that $\mathbb{V}_w \neq \emptyset$). Let $w_1, w_2 \in \mathbb{W}$. For any $w_1, w_2 \in \mathbb{W}$, we have

$$(w_1 \leq_{\mathbb{W}} w_2 \wedge w_2 \leq_{\mathbb{W}} w_1) \implies (\mathbb{V}_{w_2} = \mathbb{V}_{w_1}) \tag{13}$$

The image \mathcal{F} can also be seen as a function $\mathcal{F} : \Omega \rightarrow \mathbb{W}$. For any $w \in \mathbb{W}$, we have (see Eq. (1))

$$A_w(\mathcal{F}) = \{\mathbf{x} \in \Omega \mid w \leq_{\mathbb{W}} \mathcal{F}(\mathbf{x})\} = \{\mathbf{x} \in \Omega \mid \mathcal{F}(\mathbf{x}) \in \mathbb{V}_w\} \tag{14}$$

Now, let us consider two distinct values $w_1, w_2 \in \mathbb{W}$ such that $w_1 \leq_{\mathbb{W}} w_2$ and $w_2 \leq_{\mathbb{W}} w_1$. From Eqs. (13 and 14) it follows that $\mathcal{C}[A_{w_1}(\mathcal{F})] = \mathcal{C}[A_{w_2}(\mathcal{F})]$. In other words, relaxing the antisymmetry property to define a preorder instead of an order is useless, since it leads to define many times the same nodes which are modeled once in Θ (see Eq. (2)). We can then assume without loss of generality that $\leq_{\mathbb{W}}$ is a hierarchical order.

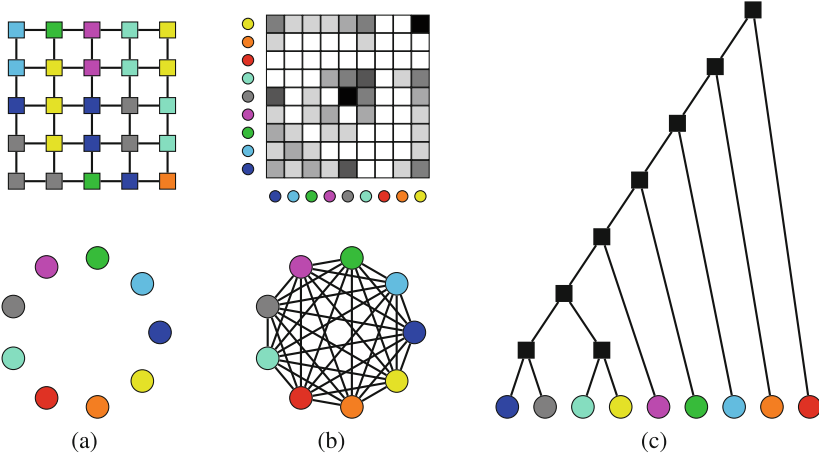


Fig. 6. Illustration of the construction of an order on a value set (see Sect. 5.2). (a) Top: an image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$. The set Ω is equal to $\llbracket 0, 4 \rrbracket^2$ and is endowed with an adjacency relation \sim corresponding to the 4-adjacency. Bottom: the set \mathbb{V} composed of 9 values, without initial ordering. (b) Top: the co-occurrence matrix of the image \mathcal{F} , used as a priority function $\delta_{\sim_{\mathbb{V}}} : \sim_{\mathbb{V}} \rightarrow \mathbb{N}$. Bottom: the set \mathbb{V} is endowed with an adjacency $\sim_{\mathbb{V}}$. (c) The binary partition tree of $(\mathbb{V}, \sim_{\mathbb{V}})$ induced by $\delta_{\sim_{\mathbb{V}}}$. This binary partition tree defines a hierarchical order \leq on an enriched set of values \mathbb{W} where the values of \mathbb{V} are the maximal elements.

Although \mathbb{W} may be of arbitrary size, we now observe that it is sufficient to define some sets \mathbb{W} that may not be larger than twice the size of \mathbb{V} . Let us set $\mathbb{V}_{\mathbb{W}} = \{\mathbb{V}_w \mid w \in \mathbb{W}\} \subseteq 2^{\mathbb{V}}$. We define the equivalence relation \equiv on \mathbb{W} by

$$(w_1 \equiv w_2) \iff (\mathbb{V}_{w_1} = \mathbb{V}_{w_2}) \tag{15}$$

Let $w \in \mathbb{W}$. The equivalence class $[w]_{\equiv}$ is totally ordered by $\leq_{\mathbb{W}}$. We set $\widehat{w} = \bigvee^{\leq_{\mathbb{W}}} [w]_{\equiv}$. We note $\widehat{\mathbb{W}} = \{\widehat{w} \mid w \in \mathbb{W}\}$. Let $w_1, w_2 \in \mathbb{W}$. We have

$$(w_1 \leq_{\mathbb{W}} w_2) \implies (\widehat{w}_1 \leq_{\widehat{\mathbb{W}}} \widehat{w}_2) \quad (16)$$

In other words, there is a homomorphism from $(\mathbb{W}, \leq_{\mathbb{W}})$ to $(\widehat{\mathbb{W}}, \leq_{\widehat{\mathbb{W}}})$ where $\leq_{\widehat{\mathbb{W}}}$ is the restriction of $\leq_{\mathbb{W}}$ to $\widehat{\mathbb{W}}$.

By construction, the set $\widehat{\mathbb{W}}$ is in bijection with $\mathbb{V}_{\mathbb{W}}$. Since $\mathbb{V} = \bigvee^{\leq_{\mathbb{W}}} \mathbb{W}$, we also have $\mathbb{V} \subseteq \widehat{\mathbb{W}}$. Following Eq. (2), we set $\Theta_{\mathbb{W}} = \bigcup_{w \in \mathbb{W}} \mathcal{C}[A_w(\mathcal{F})]$ and $\Theta_{\widehat{\mathbb{W}}} = \bigcup_{w \in \widehat{\mathbb{W}}} \mathcal{C}[A_w(\mathcal{F})]$. We have $(\Theta_{\mathbb{W}}, \subseteq) = (\Theta_{\widehat{\mathbb{W}}}, \subseteq)$. It follows that the two associated multivalued component trees are equal. As a conclusion, instead of using a set \mathbb{W} arbitrarily large and potentially infinite, a same multivalued component tree is obtained by considering the set $\widehat{\mathbb{W}}$, which is finite. Indeed, from the definition of \equiv , $\widehat{\mathbb{W}}$ is in bijection with $\mathbb{V}_{\mathbb{W}} \subseteq 2^{\mathbb{V}}$. We even have a stronger result, since the bijection between $\widehat{\mathbb{W}}$ and $\mathbb{V}_{\mathbb{W}}$ induces an isomorphism between $(\widehat{\mathbb{W}}, \leq_{\widehat{\mathbb{W}}})$ and $(\mathbb{V}_{\mathbb{W}}, \subseteq)$.

Let us now focus on the nature of the Hasse diagram of $(\mathbb{V}_{\mathbb{W}}, \subseteq)$. The inclusion \subseteq on $\mathbb{V}_{\mathbb{W}}$ is a hierarchical order. The Hasse diagram $(\mathbb{V}_{\mathbb{W}}, \triangleleft)$ is, in particular, a partition tree. The fact that $\mathbb{V} = \bigvee^{\leq_{\mathbb{W}}} \mathbb{W}$ implies that $\{\{v\} \mid v \in \mathbb{V}\} = \Delta \subseteq \mathbb{V}_{\mathbb{W}}$. It follows that $(\mathbb{V}_{\mathbb{W}}, \triangleleft)$ is a total partition tree. A corollary of this property is that $|\mathbb{V}_{\mathbb{W}}| < 2 \cdot |\mathbb{V}|$.

To conclude on this analysis, it appears that for building a hierarchical order $\leq_{\mathbb{W}}$ on a superset \mathbb{W} of \mathbb{V} so that the elements of \mathbb{V} be the maximal elements of $\leq_{\mathbb{W}}$, i.e. $\mathbb{V} = \bigvee^{\leq_{\mathbb{W}}} \mathbb{W}$, the most simple, yet general solution is to build a total partition tree from the initial, finest partition of \mathbb{V} , namely $\{\{v\} \mid v \in \mathbb{V}\}$. This can be done by building a (binary) partition tree, following the standard construction algorithms proposed in [23]. To this end, it is only required that \mathbb{V} be endowed with:

- an adjacency relation $\curvearrowright_{\mathbb{V}}$, allowing to map a graph structure on \mathbb{V} ;
- a priority function $\delta_{\curvearrowright_{\mathbb{V}}} : \curvearrowright_{\mathbb{V}} \rightarrow \mathbb{N}$, allowing to determine the couples of nodes to be merged in priority.

Example. Let us consider an image $\mathcal{F} : \Omega \rightarrow \mathbb{V}$. The support Ω is endowed with an adjacency relation \curvearrowright . The value space \mathbb{V} is endowed with the adjacency relation $\curvearrowright_{\mathbb{V}}$ so that $\mathfrak{G}_{\Delta} = (\mathbb{V}, \curvearrowright_{\mathbb{V}})$ is an irreflexive complete graph (i.e. $\forall u, v \in \mathbb{V}, u \neq v \Leftrightarrow u \curvearrowright_{\mathbb{V}} v$). We define the co-occurrence matrix [7] of the image \mathcal{F} . This matrix $\mathcal{M} = (m_{u,v})_{u,v \in \mathbb{V}}$ is of dimension $|\mathbb{V}| \times |\mathbb{V}|$. For each couple $(u, v) \in \mathbb{V} \times \mathbb{V}$, it is defined by $m_{u,v} = |\{(\mathbf{x}, \mathbf{y}) \in \Omega \times \Omega \mid \mathbf{x} \curvearrowright \mathbf{y} \wedge \mathcal{F}(\mathbf{x}) = u \wedge \mathcal{F}(\mathbf{y}) = v\}|$. We define the priority function $\delta_{\curvearrowright_{\mathbb{V}}} : \curvearrowright_{\mathbb{V}} \rightarrow \mathbb{N}$ so that for any $(u, v) \in \curvearrowright_{\mathbb{V}}$, i.e. for any $u \curvearrowright_{\mathbb{V}} v$, we have $\delta_{\curvearrowright_{\mathbb{V}}}((u, v)) = m_{u,v}$. Based on the above discussion, the partition-tree $\mathfrak{T}_{\Delta} = (\Theta_{\Delta}, \triangleleft_{\Delta})$ built [23] from $(\mathfrak{G}_{\Delta}, \delta_{\curvearrowright_{\mathbb{V}}})$ defines the Hasse diagram of a hierarchical order $\leq_{\mathbb{V}}$ on \mathbb{V} . This example is illustrated in a simplified version in Fig. 6. From \mathcal{F} and $\leq_{\mathbb{V}}$, it is then possible to build the multivalued component tree of \mathcal{F} induced by its co-occurrence matrix.

6 Conclusion

In this article, we have provided new algorithmic tools for building the multivalued component tree, but also for designing hierarchical orders on sets of values, which is a required condition of images to be modeled via multivalued component trees. In previous works [10], it had already been observed that the multivalued component tree could be efficiently involved for processing label images, especially on the context of hierarchical classification. Recent advances in the study of component trees have shed light on their links with persistent homology [15], and more generally their ability to model high-level topological information. In this context, component trees are being increasingly considered as relevant topological data descriptors to be embedded in deep-learning frameworks, e.g. for the design of loss functions [19] or to model the image structure information in self-supervised learning [28]. The contributions proposed in this article allow to efficiently handle component trees not only on grey-level images, but more generally on any multivalued images endowed with a hierarchical order. This generalization paves the way to the involvement of the (multivalued) component trees in various computer vision tasks (in particular based on deep-learning) especially in the context of multivalued data, which is for instance the case in semantic segmentation.

References

1. Blin, N., Carlinet, E., Lemaitre, F., Lacassagne, L., Géraud, T.: Max-tree computation on GPUs. *IEEE Trans. Parallel Distrib. Syst.* **33**, 3520–3531 (2022)
2. Breen, E.J., Jones, R.: Attribute openings, thinnings, and granulometries. *Comput. Vis. Image Underst.* **64**, 377–389 (1996)
3. Carlinet, E., Géraud, T.: A comparative review of component tree computation algorithms. *IEEE Trans. Image Process.* **23**, 3885–3895 (2014)
4. Gazagnes, S., Wilkinson, M.H.F.: Distributed connected component filtering and analysis in 2D and 3D tera-scale data sets. *IEEE Trans. Image Process.* **30**, 3664–3675 (2021)
5. Götz, M., Cavallaro, G., Géraud, T., Book, M., Riedel, M.: Parallel computation of component trees on distributed memory machines. *IEEE Trans. Parallel Distrib. Syst.* **29**, 2582–2598 (2018)
6. Guigues, L., Cocquerez, J.P., Le Men, H.: Scale-sets image analysis. *Int. J. Comput. Vision* **68**, 289–317 (2006)
7. Haralick, R.M., Shanmugam, K.S., Dinstein, I.: Textural features for image classification. *IEEE Trans. Syst. Man Cybern.* **3**, 610–621 (1973)
8. Jones, R.: Connected filtering and segmentation using component trees. *Comput. Vis. Image Underst.* **75**, 215–228 (1999)
9. Kiran, B.R., Serra, J.: Global-local optimizations by hierarchical cuts and climbing energies. *Pattern Recogn.* **47**, 12–24 (2014)
10. Kurtz, C., Naegel, B., Passat, N.: Connected filtering based on multivalued component-trees. *IEEE Trans. Image Process.* **23**, 5152–5164 (2014)
11. Monasse, P., Guichard, F.: Scale-space from a level lines tree. *J. Vis. Commun. Image Represent.* **11**, 224–236 (2000)

12. Najman, L., Schmitt, M.: Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **18**, 1163–1173 (1996)
13. Ouzounis, G.K., Wilkinson, M.H.F.: Mask-based second-generation connectivity and attribute filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**, 990–1004 (2007)
14. Passat, N., Kenmochi, Y.: A topological tree of shapes. In: DGMM, pp. 221–235 (2022)
15. Passat, N., Mendes Forte, J., Kenmochi, Y.: Morphological hierarchies: a unifying framework with new trees. *J. Math. Imaging Vis.* **65**, 718–753 (2023)
16. Passat, N., Naegel, B., Kurtz, C.: Component-graph construction. *J. Math. Imaging Vis.* **61**, 798–823 (2019)
17. Passat, N., Naegel, N.: Component-trees and multivalued images: structural properties. *J. Math. Imaging Vis.* **49**, 37–50 (2014)
18. Perret, B., Lefèvre, S., Collet, C., Slezak, É.: Hyperconnections and hierarchical representations for grayscale and multiband image processing. *IEEE Trans. Image Process.* **21**, 14–27 (2012)
19. Perret, B., Cousty, J.: Component tree loss function: definition and optimization. In: DGMM, pp. 248–260 (2022)
20. Randrianasoa, J.F., Kurtz, C., Desjardin, E., Passat, N.: Binary partition tree construction from multiple features for image segmentation. *Pattern Recogn.* **84**, 237–250 (2018)
21. Ronse, C., Agnus, V.: Morphology on label images: flat-type operators and connections. *J. Math. Imaging Vis.* **22**, 283–307 (2005)
22. Rosenfeld, A.: Digital topology. *Am. Math. Mon.* **86**, 621–630 (1979)
23. Salembier, P., Garrido, L.: Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE Trans. Image Process.* **9**, 561–576 (2000)
24. Salembier, P., Oliveras, A., Garrido, L.: Anti-extensive connected operators for image and sequence processing. *IEEE Trans. Image Process.* **7**, 555–570 (1998)
25. Salembier, P., Serra, J.: Flat zones filtering, connected operators, and filters by reconstruction. *IEEE Trans. Image Process.* **4**, 1153–1160 (1995)
26. Salembier, P., Wilkinson, M.H.F.: Connected operators. *IEEE Signal Process. Mag.* **26**, 136–157 (2009)
27. Soille, P.: Constrained connectivity for hierarchical image decomposition and simplification. *IEEE Trans. Pattern Anal. Mach. Intell.* **30**, 1132–1145 (2008)
28. Tang, Q., Du, B., Xu, Y.: Self-supervised learning based on max-tree representation for medical image segmentation. In: IJCNN, pp. 1–6 (2022)



Sketch2Seg: Sketch-Based Image Segmentation with Pre-trained Diffusion Model

Xin Dai, Haoge Deng, Ke Li, and Yonggang Qi^(✉)

Beijing University of Posts and Telecommunications, Beijing, China
{dxin1111,denghaoge,like1990,qiyg}@bupt.edu.cn

Abstract. The text-to-image diffusion models have been applied to image segmentation, demonstrating the potential of diffusion models in segmentation. However, texts often struggle to accurately describe objects, particularly when it comes to fine-grained details. Sketches, on the other hand, can address the issue to some extent. We observed that the intermediate features of the diffusion model guided by sketch contain more effective semantic information for segmentation compared to those guided by text. Therefore, we propose Sketch2Seg, a sketch-based image segmentation method with a diffusion model. By extracting intermediate features of a sketch-to-image diffusion model, only a simple pixel classifier needs to be trained. Quantitatively, our method reaches 78.47% and 81.07% mIOU on the PASCAL VOC and SketchySeg datasets on zero-shot setups, respectively. To investigate fine-grained sketch segmentation and detection, we contribute the SketchyCOCOseg dataset which contains segmentation annotations for images corresponding to the Sketchy-COCO dataset. Our code is available [here](#).

Keywords: Sketch-based image segmentation · Diffusion model · Dataset

1 Introduction

Diffusion models are employed for state-of-the-art image generation. Simultaneously, they have shown excellent results in various segmentation tasks [1–4]. Especially some researches on text-to-image diffusion models [5, 6], extends the potential for text-assisted segmentation. However, texts are not always the most appropriate way to describe the exact object people want to segment, especially when it comes to fine-grained object details such as the location, shape, and pose [7].

In recent years, sketch, as a complementary modality to text, has been explored broadly due to the ubiquitous nature of touchscreens [8, 9]. The common perspective is that sketch is preferable when words become impractical for conveying a specific concept. Sketch conveys many words [7], which provides an efficient and precise description (e.g. shape, pose, style) than text [10, 11].

Additionally, sketch is a special kind of visual data, inherently has a smaller gap with natural images than text, contributes to the model’s understanding of segmented regions. Hence, we intend to employ sketch as an alternative to text for segmentation.

Inspired by aranchuk’s excellent work [2], we perform K-means clustering on the intermediate features of the conditional diffusion model. We compared three different conditions: sketch, text and no condition. As shown in Fig. 1, it can be observed that the intermediate features of the pre-trained diffusion model already possess certain semantic information. Moreover, the features guided by sketch can generate more optimal clustering results than text. The insight arises from this, sketch is effective guiding condition in segmentation tasks.

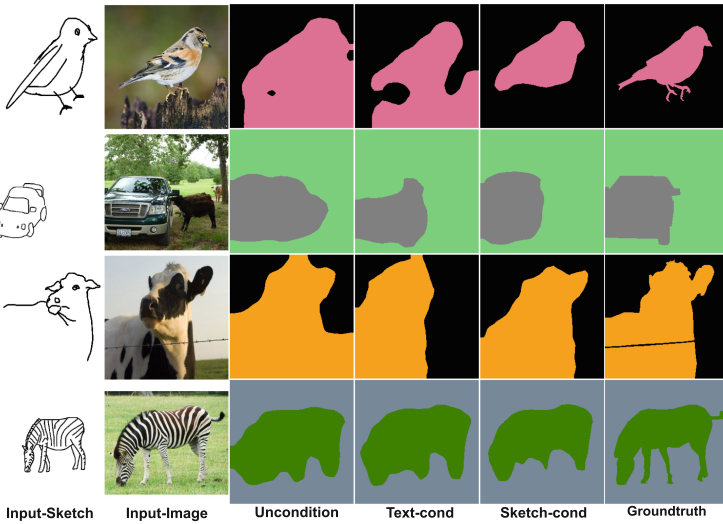


Fig. 1. K-Means ($k=2$) clustering of different conditional diffusion models’ intermediate features. From top to bottom, the text inputs are: “a photo of a bird”, “a green car on the left”, “a photo of a cow”, “a zebra grazing on the grass” (Color figure online).

In this paper, for the first time, diffusion model is exploited for sketch-based image segmentation. Specifically, we utilize ControlNet [12] and Stable Diffusion [13] for sketch and image feature extraction, respectively. These features are then employed to train a simple pixel classifier, achieving segmentation of target objects and background. Sketch-to-image diffusion model [14] is a diffusion generation model that generates natural images conditioned on sketch as input. Our model ingeniously utilizes a pre-trained sketch-to-image diffusion model, eliminating the need for extensive training data. In addition to few-shot training, owing to the ability of sketch to convey abstract concepts, we further dictate our model works in a zero-shot manner for a general-purpose [15].

We conducted category-level and instance-level segmentation on a subset of PASCAL VOC 2012 [16] and SketchySeg [8] dataset, respectively. The results

demonstrate that our approach outperforms existing works. During our research, we discovered the absence of dataset for fine-grained segmentation setting [15, 17]. Therefore, following the approach taken by chowdhury et al. [15] in sketch-based object detection, we contributed a dataset suitable for fine-grained sketch image segmentation.

In summary, our contributions are (i) for the first time, we propose a sketch-based image segmentation method with diffusion models. (ii) our approach requires few data for training, and achieves the best performance in a zero-shot manner. (iii) we contribute a granular dataset for fine-grained sketch-image segmentation and detection.

2 Related Work

Sketch-based Image Segmentation. Sketch-based image segmentation differs from traditional dense prediction tasks such as semantic segmentation and instance segmentation. It regards sketch as a description of the desired concept to segment and separate objects represented by sketch from the background in the image. The existing work extends traditional segmentation networks to introduce the concept of sketch. Sketch-a-segmenter [8] first proposed the visual task, they extended the DeepLabv3+ [18] segmentation network to include a Hypernetwork [19] that introduce the sketch and synthesizes the weights of DeepLabv3+ classifier. SGOL [20] proposed a variant of the DETR [21] network that explores sketch-guided instance segmentation through an encoder-decoder transformer architecture.

Diffusion Model for Image Segmentation. Diffusion model have recently shown remarkable potential in various segmentation tasks [5, 6, 22]. Image segmentation was transformed into mask generation by [22–24]. Text-to-image diffusion model were exploited to generate cross-attention maps during the denoising process, which were then used for target segmentation [25]. Aranchuk et al. utilized the intermediate features generated by the diffusion model for segmentation [2]. To the best of our knowledge, our method is the first to combine sketch and diffusion models for image segmentation.

3 Methodology

3.1 Problem Definition

Referring to Sketch-a-Segmentor [8], we use sketches as a visual representation concept for the segmentation part, segmenting certain types of objects in an image. To train a classifier, we use N data tuples $\{(\mathbf{p}, \mathbf{s}, \mathbf{m})\}_{i=1}^N$ about photo \mathbf{p} , sketch \mathbf{s} and segmentation mask \mathbf{m} . Formally, our segmentation model can be expressed as

$$\mathbf{m} = \mathcal{S}_{\Phi}(\mathcal{F}_{\Theta}(\mathbf{p}, \mathbf{s})) \quad (1)$$

where $\mathcal{F}_{\Theta}(\cdot; \cdot)$ is a feature extractor of image and sketch, $\mathcal{S}_{\Phi}(\cdot)$ is a pixel classifier with parameters Φ .

3.2 Diffusion Model

The diffusion model first adds Gaussian noise gradually to the picture x_0 through a forward process to get the noisy sample x_t , which finally makes the image become completely Gaussian noise $x_T \sim \mathcal{N}(0, \mathbf{I})$. Formally, the forward diffusion process can be

$$q(x_t | x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t \mathbf{I}\right) \quad (2)$$

where β_1, \dots, β_t is a fixed variance schedule. The diffusion model defines each addition of noise as a Markov process, where upon the intermediate samples of the addition of noise can be obtained directly from the original image x_0

$$q(x_t | x_0) := \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1 - \bar{\alpha}_t) \mathbf{I}\right) \quad (3)$$

where $\alpha_t := 1 - \beta_t, \bar{\alpha}_t := \prod_{s=1}^t \alpha_s$.

The reverse process is also defined as a Markov Gaussian transformation,

$$p_\theta(x_{t-1} | x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (4)$$

where the mean of the Gaussian distribution μ_θ needs to be predicted using a neural network, the variance Σ_θ can be a parameter that is considered fixed or learnable. However, during the practice of diffusion modeling, it has been found that it is more effective to predict the noise ε_θ in the denoising process than to predict the mean directly, and it can be shown that the μ_θ is a linear combination of ε_θ and x_t

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{\beta_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_\theta(x_t, t) \right) \quad (5)$$

3.3 Feature Extraction

The prevalent diffusion-based image generative models [12, 13, 26, 27] typically use a UNet architecture to learn the denoising process. As shown in Fig. 2, We exploit pre-trained ControlNet [12] to encode sketch, facilitating the integration of sketch information with the images to be segmented. ControlNet is a neural network that controls a pretrained image diffusion model, which replicates the structure and parameters of UNet’s encoder, and different modules are connected to each other using a 1×1 convolutional layer with an initialisation parameter of zero. At every step of the de-noising process, diffusion model use the sketch input to infer the de-noising direction of the noising input image.

As shown in [2, 6], the feature map output of the UNet can be regarded as rich and dense feature for segmentation. We extract certain output of the UNet’s decoders as feature fusion as sketch and image, which leading pixel classifier to distinguish between the object to be segmented and other objects.

For a given real image $x_0 \in \mathbf{R}^{H \times W \times 3}$ and the segment query sketch $s_0 \in \mathbf{R}^{H \times W \times 3}$, we compute reverse process of an input image and sketch at certain de-noising step t through the diffusion model to extract its visual representation.

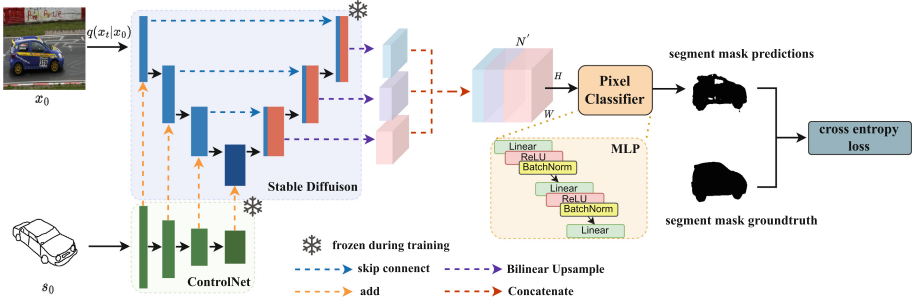


Fig. 2. Overview and training pipeline of the proposed method. We first corrupt the input image x_0 according to $q(x_t|x_0)$ to obtain x_t , then x_t and sketch s_0 were fed into Stable diffusion and ControlNet, respectively. Then we extract the intermediate features of the certain blocks of the decoder, upsampling to the image resolution and concatenating them. The pixel-level representations are used to train a MLP with cross entropy loss.

Specially, we first corrupt x_0 by adding Gaussian noise according to Eq. 3 at several certain time step t . The noisy x_t is input of the UNet and s_0 is fed into ControlNet. The outputs of each module of ControlNet are added to UNet’s encoder. During the reverse process, we extract specific blocks’ intermediate features and upsample to $H \times W$ with bilinear interpolation. All the feature maps concatenated allows treating them as pixel-level representation of input x_0 and s_0 . We denote the joint feature as $F_0 \in \mathbf{R}^{H \times W \times N'}$, which N' being the dimension of the high-dimensional feature representation of a single pixel.

3.4 Pixel Classifier

Reshape F_0 into a two-dimensional vector $V_0 \in \mathbf{R}^{HW \times N'}$, and for each pixel representation $f_i \in \mathbf{R}^{N'}$ in V_0 , we train a multi-layer perceptrons(MLP) to classify them. The MLP consists of multiple Linear, ReLU and Batch-Normalization layers. When predicting the categories, the *softmax* function is used to take the category with the highest probability as the predicted category.

$$\hat{m}_i = \text{softmax}(MLP(f_i)) \quad (6)$$

where $\hat{m}_i \in \{0, 1\}$, denotes the category of the i th pixel.

3.5 Object Function

We train the network to binary classify each pixel of the image to get a binary (foreground and background) segmentation map. We use the cross-entropy function as an objective function to minimize the cross-entropy between the predicted and true values during training.

$$\mathcal{L} = - \sum \mathbf{m}_{p(i,j)} \log(\hat{\mathbf{m}}_{p(i,j)}) \quad (7)$$

where $p_{(i,j)}$ denotes the pixel in row i , column j . $\mathbf{m}_{p_{(i,j)}}$ is the true labeling of the corresponding pixel, and $\hat{\mathbf{m}}_{p_{(i,j)}}$ is the label of the category predicted by the model.

Table 1. Categories of the Training and Testing

Train-Category(10)	Test-Category(5)
dog,wine_bottle,horse,sheep,airplane	car,cat,chair
bicycle,sailboat,songbird,table,coach	cow,motorcycle

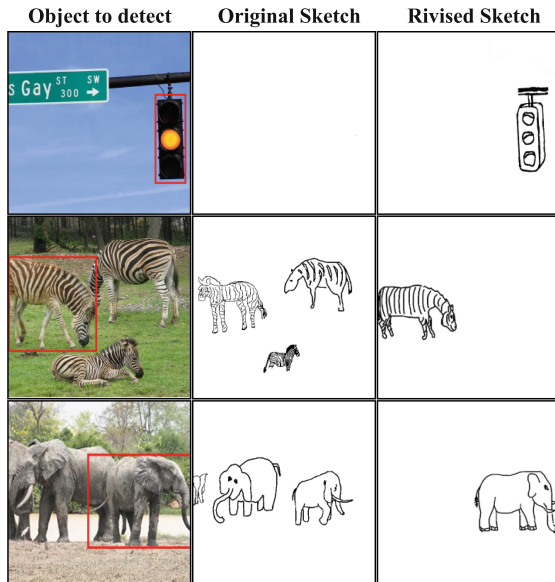


Fig. 3. Examples about revised fine-grained object detection dataset SketchyCOCOseg

4 Experiment

To probe the limits of sketch segmentation, we conducted experiments at three granularities: category, single-object instance, and multi-object fine-grained levels, using unique datasets for each. For fine-grained segmentation without comparative SketchyCOCO results, we converted the task to object detection via segmentation masks' minimum bounding rectangles. All experiments were evaluated in a zero-shot setting for the training and testing classes are completely disjoint.

4.1 Experiment Setting and Datasets

Datasets. For category-level image segmentation, we select photos from 15 categories in the PASCAL VOC 2012 dataset following [8], the sketches come from Sketchy dataset [28]. Specifically, the categories chosen are listed in Table 1. For instance-level image segmentation, we choose the SketchySeg dataset proposed by [8]. SketchySeg dataset contains 15 annotated categories as listed in Table 1. For sketch-based fine-grained image detection, 1225 sketch-image pairs from SketchyCOCO [29] with at least one foreground object are chosen following [15].

Correct Fine-Grained Segmentation Dataset. Since most of the sketches in the SketchyCOCO dataset are obtained by selecting the foreground from the scene sketch, which results in some of the sketches being almost a blank map or the sketches have no correspondence with the objects to be detected. We corrected the sketches in this dataset.

Our method is: in the SketchyCOCO dataset, the position and pose of the object to be detected guide the manual selection of sketches from the Sketchy dataset. Sketches that exhibit a similar pose to the object in the image are chosen from sketches of the same category of object. These selected sketches are then used as pairing data. Image processing software is employed to align the position and size of the sketches with the corresponding objects in the images. In the new dataset, there are a total of 1225 pairs of data across 14 categories¹. Each photo in the dataset has a corresponding fine-grained sketch and annotated segmentation mask. We have named this new dataset SketchyCOCOseg. Some examples are shown in Fig. 3.

Implementation Details. We use PyTorch 1.12 and run our experiment on 4 Tesla T4 gpus. The image and sketch crop size in the experiment is 256×256 , the UNet model is Stable Diffusion [13]. The ControlNet and Stable Diffusion load pre-trained parameters scribble2image². The UNet decoder blocks we choose are $B = \{5, 7, 8, 11\}$ and the denoising steps are $t = \{50, 100, 200\}$ of the reverse diffusion process.

Evaluation Protocol. For image segmentation, we use the standard mean Inter-section-Over-Union(mIOU) and pixel accuracy for evaluation. The image pixels belonging to the category of sketch are regraded as foreground and the rest as background. For object detection, we measure $AP_{.3}$, $AP_{.5}$ and $AP_{.7}$ that computes the average precision (AP) at IoU values 0.3, 0.5, and 0.7 following [15].

¹ airplane,bicycle,car,cat,cow,dog,elephant,fire_hydrant,giraffe,horse,motorcycle,sheep, traffic_light,zebra.

² <https://github.com/llyasviel/ControlNet>.

4.2 Competitors

Image Segmentation

DeepLabv3+ [18]: A general image segmentation model. In our experiment, only the photos are fed into DeepLabv3+ for generic binary foreground/background segmentation.

OSLSM [30]: For training, one masked photo in the Sketchy dataset are exploited for parameter generation.

DDPM-Seg [2]: A diffusion model based baseline. The Stable Diffusion [13] are used for feature extraction and the category number is two. We retrained the classifier on our own dataset to evaluate performance.

DeepLabv3+sketch [8]: Both image and sketch are depth concatenated and fed into the network for image segmentation map prediction.

Hyper-DeepLabv3+ Wordvec [8]: A language-based zero-shot learning architecture. One 300-d vector which extracted for each category by using the word2vec model pre-trained on the Google News corpus [32] are exploited to guide classification layer parameters synthesis.

Hyper-DeepLabv3+ sketch [8]: Sketch-based segmentation model, which takes sketch as input and generate corresponding weights for segmentation.

SAM [31]: A general segmentation model, we take the outer rectangle of the input sketch as input to get the segmentation mask.

DDPM-Seg + text: A language-based comparative model, which encodes textual descriptions of the objects in the image using a pre-trained CLIP model, providing guidance to the pixel classifier.

Table 2. Category-Level Segmentation Result

Method	mIOU(%)	Pixel Accuracy(%)
DeepLabv3+	65.31	83.52
OSLSM	65.40	84.36
DeepLabv3+ sketch	67.54	86.25
Hyper-DeepLabv3+ Wordvec	70.84	89.30
Hyper-DeepLabv3+ sketch	73.35	90.24
DDPM-Seg	74.54	89.51
Sketch2Seg	78.47	90.50

Fine-Grained Object Detection

WSDDN [33]: Reimagining object detection as a region classification task using the Multiple Instance Learning (MIL) paradigm. To incorporate a query sketch into the WSDDN framework, we employ cross-attention in conjunction with RoI pooled features, subsequently employing a binary classifier for detection.

OICR [34]: Building upon the foundation of WSDDN, improvements are introduced through iterative MIL to refine the initial prediction scores.

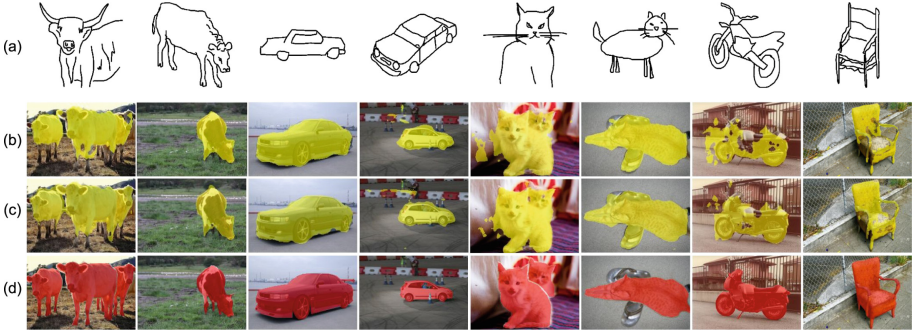


Fig. 4. Results of Category-level Segmentation on Pascal VOC and Sketchy. (a) Input sketch for sketch2seg model, (b) Segmentation results of DDPM-Seg, (c) Segmentation results of DDPM-Seg, (d) Ground-truth segmentation map.

PCL [35]: Multiple positive instances are generated through clustering, and each cluster result is assigned the corresponding object class label.

ICMWS [36]: The network are forced to look in the surrounding context regions by dropping the most discriminative parts.

Sketch-Detect [15]: A sketch-aware detector, which applies a novel prompt learning setup to marry CLIP [37] and sketch-based image retrieval(SBIR).

Table 3. Instance-Level Segmentation Result

Method	mIOU(%)	Pixel Accuracy(%)
DeepLabv3+	74.35	88.17
DeepLabv3+ sketch	76.78	88.77
Hyper-DeepLabv3+ sketch	80.48	90.92
SAM	79.67	90.17
DDPM-Seg	77.94	88.39
DDPM-Seg + text	79.93	91.24
Sketch2seg	81.07	91.33

4.3 Experiment Result

Sketch-Based Category-Level Segmentation

Settings: For the 10 training categories, 5 images are randomly selected for each category, and sketches of the same category are randomly selected for each image as training data. When testing, for each picture in the test set, a sketch of the same category is randomly selected. The test experiment is repeated three times and the results are averaged.



Fig. 5. Result of Instance-level Segmentation on SketchySeg. (a) Input sketch for sketch2seg model. (b) Segmentation results of DDPM-Seg. (c) Segmentation results of sketch2seg + text. (d) Segmentation results of sketch2seg + sketch. (e) Ground-truth segmentation map

Results: The results are shown in Table 2, from which we can observe: (i) The DDPM-Seg outperforms Deeplabv3+ and OSLSM when no prior information is available regarding auxiliary segmentation. (ii) When applied to novel categories, the binary foreground segmentation model can somewhat predict foreground versus background, despite its original training on different categories. (iii) Category information proves to be beneficial in enhancing segmentation accuracy, with sketches demonstrating a higher level of efficacy for segmentation when compared to word vectors. (iv) In the zero-shot scenario, our proposed approach attains superior results compared to existing methods. Some qualitative results are shown in Fig. 4. Compared with DDPM-seg, Sketch2Seg only uses sketches of the same category, which can obtain more accurate segmentation edges and reduce segmentation holes.

Sketch-Based Instance-Level Segmentation

Settings: The train/test category split is the same as that for category-level segmentation. In the Sketchy dataset, multiple instance-level sketches were drawn for each image, for testing, we randomly selected one as the segmentation query. For the sketch2Seg+text model, we use a general prompt “a photo of a [class]” where [class] denotes the input images’ category. For instance, “a photo of a cow” for the first column in the Fig. 5.

Results: From Table 3 we can see: (i) Any description (text or sketch) of the object to segment provides improved performance compared to a generic fore-

Table 4. Fine-Grained Object Detection Result

Method	$AP_{.3}$	$AP_{.5}$	$AP_{.7}$
WSDDN	8.1	10.2	9.4
OICR	8.9	10.9	10.0
PCL	9.2	11.5	10.6
ICMWSO	10.3	11.9	10.8
Sketch-Detect	15.0	17.1	16.3
Sketch2seg(SketchCOCO)	48.9	38.4	29.6
Sketch2seg(SketchCOCOseg)	54.2	40.1	31.9

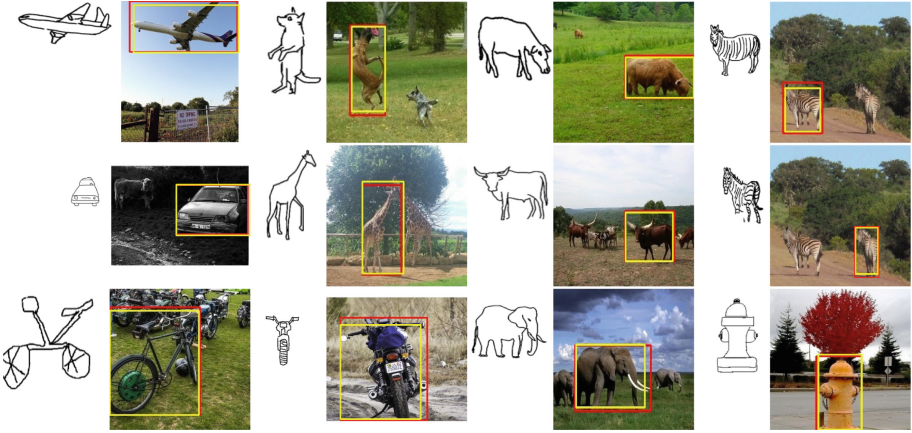


Fig. 6. Fine-Grained Object Detection with sketch query on image from Sketchy-COCO. The red box is ground-truth bounding box, the yellow box is our predict bounding box. (Color figure online)

ground segmenter in our experiment. (ii) For the objects under segmentation, the instance-level sketches yield a more substantial enhancement in segmentation results compared to textual descriptions. (iii) Despite training with a limited volume data, our proposed approach achieves state-of-the-art performance in the zero-shot setting. Figure 5 presents some qualitative results. Comparing to DDPM-Seg without any auxiliary information and sketch2seg+text, sketch2seg+sketch can obtain more accurate segmentation map, as shown in (d) row, the segmentation results have less segment cavities.

Sketch-Based Fine-Grained Object Detection

Settings: We classify the 1225 images based on the identified objects into a total of 14 categories. We randomly select 9 categories, 5 images from each category as training data. All data from the remaining 5 categories are used as test data. Repeat 5 times and the results are averaged.

Table 5. Cross-dataset Evaluation Result

Dataset	mIOU(%)	Pixel Accuracy(%)
Quickdraw	78.06	89.82
Tuberlin	79.68	90.78
Sketchy	81.07	91.13

Results: From Table 4 we can see: (i) Our method improves AP dramatically, with an 81% improvement in the metric AP_7 compared to the previous best method. (ii) Our New dataset SketchCOCOseg has improved accuracy for target detection. (iii) Our method also works for fine-grained object detection, and our method does not require multiple candidate boxes and Non-Maximum Suppression(NMS) process. Some qualitative results are shown in Fig. 6. Our model capture fine-grained object information and detect specific objects represented in the sketch.

4.4 Cross-Dataset Evaluation

One goal of our model is to handle the high variability observed in sketch [20], allowing segmentation queries across different styles without the need for retraining. Hence, we chose five test categories from the SketchySeg dataset and selected corresponding sketches from the QuickDraw [38] and Tuberlin [39] dataset to form data pairs. Our model underwent training on the Sketchy dataset and was subsequently tested directly on the two newly composed datasets. The results are shown in Table 5. Compared to the Sketchy dataset, segmentation mIOU drops slightly on both new datasets, and drops even more on the QuickDraw dataset, with mIOU dropping by 3.71%, which is to be expected since the sketches in Quickdraw are really simple and abstract. In general, our approach demonstrates the capability to perform direct segmentation across datasets.

5 Conclusion

In this article, we presented Sketch2Seg, a new approach to sketch-based image segmentation. Our approach exploits pre-trained diffusion model to represent sketch and image, requires a few annotated training data and achieves optimal performance even in zero-shot setup. In order to explore the expression of sketch in fine-grained segmentation and detection tasks, we introduce a new dataset called SketchyCOCOseg, based on SketchyCOCO. This dataset alleviates the data scarcity issue in fine-grained segmentation tasks.

Acknowledgements. This work was supported by NSFC under No. 61601042 and the Program for Youth Innovative Research Team of BUPT under No. 2023YQTD02.

References

1. Rahman, A., Valanarasu, J.M.J., Hacihaliloglu, I., Patel, V.M.: Ambiguous medical image segmentation using diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11536–11546 (2023)
2. Baranchuk, D., et al.: Label-efficient semantic segmentation with diffusion models. In: International Conference on Learning Representations (2021)
3. Amit, T., Shaharbany, T., Nachmani, E., Wolf, L.: SegDiff: image segmentation with diffusion probabilistic models. arXiv preprint [arXiv:2112.00390](https://arxiv.org/abs/2112.00390) (2021)
4. Brempong, E.A., Kornblith, S., Chen, T., Parmar, N., Minderer, M., Norouzi, M.: Denoising pretraining for semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4175–4186 (2022)
5. Pnvr, K., Singh, B., Ghosh, P., Siddiquie, B., Jacobs, D.: LD-ZNet: a latent diffusion approach for text-based image segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4157–4168 (2023)
6. Xu, J., Liu, S., Vahdat, A., Byeon, W., Wang, X., De Mello, S.: Open-vocabulary panoptic segmentation with text-to-image diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2955–2966 (2023)
7. Qi, Y., Song, Y.Z., Zhang, H., Liu, J.: Sketch-based image retrieval via siamese convolutional neural network. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 2460–2464. IEEE (2016)
8. Hu, C., Li, D., Yang, Y., Hospedales, T.M., Song, Y.Z.: Sketch-a-Segmenter: sketch-based photo segmenter generation. *IEEE Trans. Image Process.* **29**, 9470–9481 (2020)
9. Yang, R., Li, D., Hu, C., Hospedales, T., Zhang, H., Song, Y.Z.: Sketch-based video object segmentation: benchmark and analysis. arXiv preprint [arXiv:2311.07261](https://arxiv.org/abs/2311.07261) (2023)
10. Hu, R., Collomosse, J.: A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Comput. Vis. Image Underst.* **117**(7), 790–806 (2013)
11. Xu, P., Hospedales, T.M., Yin, Q., Song, Y.Z., Xiang, T., Wang, L.: Deep learning for free-hand sketch: a survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(1), 285–312 (2022)
12. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3836–3847 (2023)
13. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10684–10695 (2022)
14. Voynov, A., Aberman, K., Cohen-Or, D.: Sketch-guided text-to-image diffusion models. In: ACM SIGGRAPH 2023 Conference Proceedings (2023)
15. Chowdhury, P.N., Bhunia, A.K., Sain, A., Koley, S., Xiang, T., Song, Y.Z.: What can human sketches do for object detection?. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15083–15094 (2023)
16. Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge: a retrospective. *Int. J. Comput. Vision* **111**, 98–136 (2015)

17. Pang, K., Yang, Y., Hospedales, T.M., Xiang, T., Song, Y.Z.: Solving mixed-modal jigsaw puzzle for fine-grained sketch-based image retrieval. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10347–10355 (2020)
18. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 833–851. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_49
19. Ha, D., Dai, A., Le, Q.V.: Hypernetworks. arXiv preprint [arXiv:1609.09106](https://arxiv.org/abs/1609.09106) (2016)
20. Riba, P., Dey, S., Biten, A.F., Llados, J.: Localizing infinity-shaped fishes: sketch-guided object localization in the wild. arXiv preprint [arXiv:2109.11874](https://arxiv.org/abs/2109.11874) (2021)
21. 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
22. Dombrowski, M., Reynaud, H., Baugh, M., Kainz, B.: Foreground-background separation through concept distillation from generative image foundation models. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 988–998 (2023)
23. Chen, T., Li, L., Saxena, S., Hinton, G., Fleet, D.J.: A generalist framework for panoptic segmentation of images and videos. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 909–919 (2023)
24. Kim, B., Oh, Y., Ye, J.C.: Diffusion adversarial representation learning for self-supervised vessel segmentation. arXiv preprint [arXiv:2209.14566](https://arxiv.org/abs/2209.14566) (2022)
25. Wang, J., et al.: Diffusion model is secretly a training-free open vocabulary semantic segmenter. arXiv preprint [arXiv:2309.02773](https://arxiv.org/abs/2309.02773) (2023)
26. Nichol, A., et al.: Glide: towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint [arXiv:2112.10741](https://arxiv.org/abs/2112.10741) (2021)
27. Dhariwal, P., Nichol, A.: Diffusion models beat GANs on image synthesis. In: Advances in Neural Information Processing Systems, vol. 34, pp. 8780–8794 (2021)
28. Sangkloy, P., Burnell, N., Ham, C., Hays, J.: The sketchy database: learning to retrieve badly drawn bunnies. ACM Trans. Graph. (TOG) **35**(4), 1–12 (2016)
29. Gao, C., Liu, Q., Xu, Q., Wang, L., Liu, J., Zou, C.: SketchyCOCO: image generation from freehand scene sketches. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5174–5183 (2020)
30. Shaban, A., Bansal, S., Liu, Z., Essa, I., Boots, B.: One-shot learning for semantic segmentation. arXiv preprint [arXiv:1709.03410](https://arxiv.org/abs/1709.03410) (2017)
31. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Girshick, R.: Segment anything. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4015–4026 (2023)
32. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510–4520 (2018)
33. Bilen, H., Vedaldi, A.: Weakly supervised deep detection networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2846–2854 (2016)
34. Tang, P., Wang, X., Bai, X., Liu, W.: Multiple instance detection network with online instance classifier refinement. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2843–2851 (2017)
35. Tang, P., et al.: PCL: proposal cluster learning for weakly supervised object detection. IEEE Trans. Pattern Anal. Mach. Intell. **42**(1), 176–191 (2018)

36. Ren, Z., et al.: Instance-aware, context-focused, and memory-efficient weakly supervised object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10598–10607 (2020)
37. L Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
38. Jongejan, J., et al.: The Quick, Draw! <http://www.springer.com/lncs> (2016)
39. Eitz, M., Hays, J., Alexa, M.: How do humans sketch objects? ACM Trans. Graph. (TOG) **31**(4), 1–10 (2012)



Unsupervised Segmentation of Pulmonary Regions in 3D CT Scans Optimized Using Transformer Model

Ahmed Sharafeldeen¹, Adel Khelifi², Mohammed Ghazal³, Maha Yaghi³,
Ali Mahmoud¹, Sohail Contractor⁴, and Ayman El-Baz¹✉

¹ Department of Bioengineering, University of Louisville, Louisville, KY, USA
aselba01@louisville.edu

² Computer Science and Information Technology Department, Abu Dhabi University,
Abu Dhabi, UAE

³ Electrical, Computer and Biomedical Engineering Department, Abu Dhabi
University, Abu Dhabi, UAE

⁴ Department of Radiology, University of Louisville, Louisville, KY, USA

Abstract. This paper presents a new automated unsupervised segmentation system to accurately delineate the pulmonary region in 3D computed tomography (CT) scans. It operates on a multi-dimensional joint probability model, which leverages a deep learning-based transformer model to optimize the log-likelihood of that probabilistic model. This probability model integrates appearance probability models, that represent various radiodensity distributions in both lung and chest areas within the 3D CT volume using linear combination of Gaussian (LCG), along with their spatial probability model, generated based on a 3D Markov Gibbs random field (MGRF), with potentials estimated analytically. Finally, the generated segmentation is further refined using a transformer model to maximize the log-likelihood of a given region. The proposed method's efficacy is assessed by analyzing 3D chest scans of 28 patients diagnosed with varying severities of COVID-19. Four metrics are employed for evaluation, namely, Dice similarity coefficient (DSC), overlap coefficient, Hausdorff distance (HD), and absolute volume difference (AVD). The proposed system demonstrates outstanding performance, with scores of $95.60 \pm 1.37\%$, $91.61 \pm 2.51\%$, 6.56 ± 2.68 , and 6.23 ± 3.52 , respectively. These findings underscore the potential of the proposed system in delineating both normal and pathological lung regions in CT images, when compared to its individual components as well as six state-of-the-art (SOTA) segmentation methods.

Keywords: LCG · Lung Segmentation · MGRF · Multi-dimensional Joint Probability · Transformer model

1 Introduction

The respiratory system, comprising the lungs and airways, plays a vital role in sustaining life by facilitating the exchange of oxygen and carbon dioxide. How-

ever, this intricate system is susceptible to a myriad of diseases, ranging from acute infections to chronic conditions, each with its unique set of challenges and implications for health. Pulmonary diseases present a broad array of symptoms, including coughing and difficulty breathing, progressing to respiratory failure and even mortality [30]. Early detection of pulmonary diseases is paramount, given their potential to progress silently and impact health profoundly. Consider conditions like lung cancer, which often develop asymptotically until reaching advanced stages, pulmonary fibrosis, characterized by progressive scarring of lung tissue, or coronavirus disease 2019 (COVID-19), which can significantly impact the respiratory system. Detecting these diseases early offers a crucial window for intervention, enhancing treatment efficacy and improving patient outcomes. Computed tomography (CT) scans, commonly employed for diagnosing pulmonary diseases, offer highly detailed visualization of the lungs, thereby enabling precise and accurate identification of respiratory conditions. To develop an accurate computer-aided diagnostic system (CAD), precise delineation of the lung region is crucial. Inaccurate segmentation of lung regions can severely compromise the effectiveness of the diagnostic process overall. Accomplishing this task presents considerable challenges, given the intricate nature of lung structures, the variability in pathological areas, and the similarity in radiodensity between these areas and the chest. Therefore, This study focuses on the development of an accurate lung segmentation system.

Several studies have introduced segmentation methods aimed at identifying both normal and pathological areas within the lungs. These methods utilize various approaches, including thresholds, shapes, deep learning models, or hybrid methods. For example, Moghaddam and Aghazadeh [22] introduced a threshold-based lung segmentation method through the utilization of Otsu thresholding method along with connected component analysis. The Bresenham method and Freeman chain-code algorithm were integrated into their system’s pipeline to identify nodules/tumors, overlooked during segmentation. Their system was evaluated on LIDC-IDRI database, achieving a Dice similarity coefficient (DSC) of $96.12 \pm 0.3068\%$. A recent study [9] utilized U-Net and V-Net for segmenting the lung and lobes, respectively. Their system achieved a DSC of $98 \pm 3\%$ and $94 \pm 6\%$, respectively. A similar study [21] utilized U-Net with DenseNet121 as a backbone for segmenting lung regions in CT images. Devi et al. [11] introduced a segmentation system based on deep learning approach designed to outline the lung area in X-ray images. This network employed U-Net as its backbone with the utilization of scale, spatial and channel attention models, achieving a DSC of 92%. A recent study by Missimer et al. [20] implemented a segmentation system aimed at outlining the lung region in magnetic resonance imaging (MRI) images. Their system preprocessed MRI images using various approaches, including morphological transformations erosion, dilation, reconstruction, and complement. Subsequently, an adaptive thresholding method was employed to segment the lung region. Finally, a connected component analysis approach was utilized to refine the segmented region. Their system achieved a DSC of 94%. In [23], authors employed DeepLabV3+ to delineate lung regions in CT images. They assessed

its effectiveness using five pretrained networks: ResNet-50, Xception, ResNet-18, Inception-ResNet-v2, and MobileNet-v2. The system’s performance was evaluated on 750 CT images, with the best lung segmentation accuracy achieved when DeepLabV3+ was combined with ResNet-18. A similar study [16] investigated the effectiveness of four deep learning-based segmentation models, including SegNet, U-Net, U-Net++, and FCN, for delineating lung regions in X-ray images. Their results showed that U-Net++ was more effective in delineating lung regions compared to the other three models. Another study [24] proposed a three-stage deep learning-based segmentation system for delineating lung regions in CT images. In the preprocessing stage, a convolution neural network (CNN) was introduced to exclude CT images devoid of lung regions. Subsequently, a U-Net was employed to delineate lung regions in CT images. During postprocessing stage, another U-Net and CNN were utilized to refine lung contour and eliminate incorrectly segmented regions, respectively. Their system was evaluated on the 3DIRCAD and ILD database, achieving a DSC of 95%. Shi et al. [29] introduced an automatic segmentation system designed to delineate pathological lung regions in CT images. Their system leveraged an active shape model (ASM) based on low-rank and sparse decomposition (LRSD) theory. Moreover, they employed a hierarchical ASM search strategy, utilizing the LRSD shape model alongside a normalized gradient-based appearance model to refine the initially generated shape, thereby enhancing the system’s efficiency against local minima. Their system was evaluated on segmenting the right lung regions in CT images, achieving a DSC of $96.13 \pm 1.54\%$. Another study [19] implemented an automatic segmentation based on wavelet transform to delineate lung regions in CT images. Initially, they proposed an image decomposition-based smoothing filter to eliminate noise from lung in CT images while retaining the integrity of their lung outlines. Subsequently, an integration of wavelet transform with morphological operations was utilized to segment lung region. Finally, a contour correction method was employed to refine the segmentation, aimed at correcting and smoothing the identified lung contours. Their system achieved an average DSC of 98.04%, when evaluated on ILDs dataset. Further elaboration on lung segmentation techniques and related literature can be found in [4, 5, 12, 14].

Despite the extensive research conducted on lung segmentation using CT scans, previous studies have notable limitations. These include inaccuracies in segmenting pathological lesions, particularly with threshold-based systems, the dependence of shape-based techniques on manual feature selection, and the substantial dataset requirements for training deep learning-based systems. To overcome these challenges, we propose an unsupervised maximum a posteriori (MAP)-based segmentation system that employs an appearance and shape probabilistic models to cluster each region (i.e., lung or chest). This model is optimized using a deep learning approach rather than stochastic optimization algorithms, aiming to maximize the likelihood of the probabilistic model.

2 Methods

Delineating the lung region is a crucial step in developing an accurate computer-aided diagnostic (CAD) system, as the precision of the segmentation system is paramount. An inaccurate segmentation system significantly impacts the accuracy of any CAD system. Therefore, the development of a precise segmentation system is essential. To achieve this, we introduce a stochastic unsupervised segmentation system, outlined in Fig. 1, with its procedural steps expounded upon in the subsequent subsections.

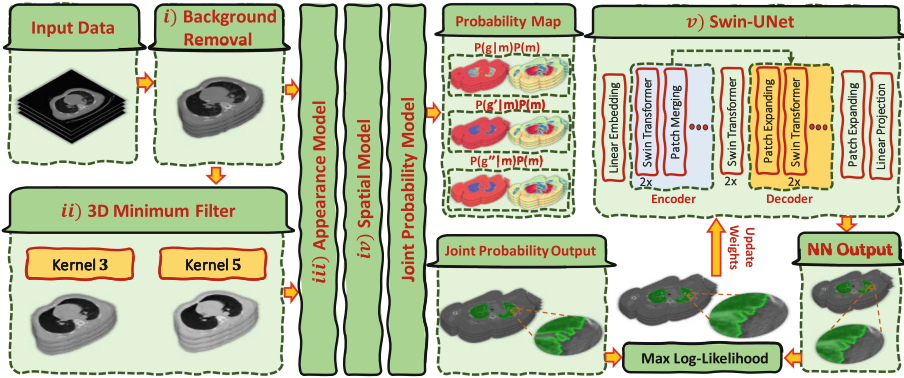


Fig. 1. A schematic visualization of the proposed unsupervised segmentation system using CT images.

2.1 Appearance Realizations

To improve the accuracy of the proposed system, various appearance realizations have been incorporated into the system pipeline. These realizations are derived using a 3D minimum filter with two different kernel sizes: 3 and 5. The rationale behind utilizing these filters is rooted in the fact that lesion regions typically exhibit higher radiodensity than lung regions. Hence, these filters serve to effectively merge these regions with the lung, thereby enhancing the lung regions homogeneity.

2.2 Joint Probability Model

To integrate the density distribution of CT data and its appearance realizations with their spatial distribution, we proposed a multi-dimensional joint probability model. Traditionally, the joint probability model relies on a single appearance realization (g) and region maps (i.e., labeled images denoted by m), that can be modeled as follows:

$$P(\mathbf{g}, \mathbf{m}) = P(\mathbf{g}|\mathbf{m})P(\mathbf{m}). \quad (1)$$

Here, $P(\mathbf{m})$ denotes the spatial distribution of a given map, while $P(\mathbf{g}|\mathbf{m})$ represents the appearance distribution of a grayscale value given a map. In order to improve the precision of this model, a multi-dimensional appearance realization ($G = g, g', g'', \dots$) is employed instead of relying on a single appearance realization. Each of these realizations (i.e., g, g', g'', \dots) represents different levels of uniformity improvement obtained using various 3D filters. In this paper, a 3D minimum filter is employed to derive multiple appearance realizations, aiming to make the lesion regions appear similar to the lung regions, thereby enhancing the appearance of lung regions. Assuming the independence of these realizations, the multi-dimensional joint probability model is defined as follows:

$$\begin{aligned} P(G, \mathbf{m}) &= P(\mathbf{G}|\mathbf{m})P(\mathbf{m}) \\ &= \underbrace{[P(\mathbf{g}|\mathbf{m}) + P(\mathbf{g}'|\mathbf{m}) + P(\mathbf{g}''|\mathbf{m})]}_{\text{Appearance Model}} P(\mathbf{m}). \end{aligned} \quad (2)$$

To regularize the distribution of a given map, Bayesian maximum a posteriori (MAP) estimation is utilized, with the aim of maximizing the log-likelihood of a map based on grayscale values, as defined below:

$$\mathbf{m}^* = \underset{\mathbf{m}}{\operatorname{argmax}} L(\mathbf{G}, \mathbf{m}), \quad (3)$$

where $L(\mathbf{g}, \mathbf{m})$ represents the log-likelihood of the proposed multi-dimensional model, estimated as follows:

$$\begin{aligned} L(\mathbf{G}, \mathbf{m}) &= \log P(\mathbf{G}, \mathbf{m}) \\ &= \log [P(\mathbf{g}|\mathbf{m}) + P(\mathbf{g}'|\mathbf{m}) + P(\mathbf{g}''|\mathbf{m})] + \log P(\mathbf{m}). \end{aligned} \quad (4)$$

2.3 Appearance Model

In order to probabilistically capture the variability in CT data and its appearance realizations, an appearance probabilistic model is introduced. This model effectively addresses the challenge posed by the different distributions of CT data resulting from different scanning protocols. It achieves this by representing the distributions (i.e., histograms) of CT data and its appearance realizations as a linear combination of Gaussian (LCG) distributions [2, 3, 26–28], as illustrated in Fig. 2. Let K denote the number of Gaussian distributions utilized to model the histogram, with at least two Gaussian components. These two Gaussian components, referred to as primary components, signify lung (i.e., $k = 1$) and chest (i.e., $k = 2$) regions. The remaining $K - 2$ Gaussian components, referred to as secondary components, are utilized to model the variations, not captured by the primary components. This probabilistic model, illustrated in Fig. 2b, is defined as follows:

$$P(h) = \underbrace{w_1\varphi(h; \theta_1) + w_2\varphi(h; \theta_2)}_{\text{primary}} + \underbrace{\sum_{k=3}^K w_k\varphi(h; \theta_k)}_{\text{secondary}}, \quad (5)$$

where w denotes the mixing weights that adjusts this probabilistic model to form a density function, with the condition that $\sum_{k=1}^K w_k = 1$. While, $\varphi(h; \theta_k)$, and $\theta_k = (\mu_k, \sigma_k)$ represent the Gaussian distribution and its parameters, respectively.

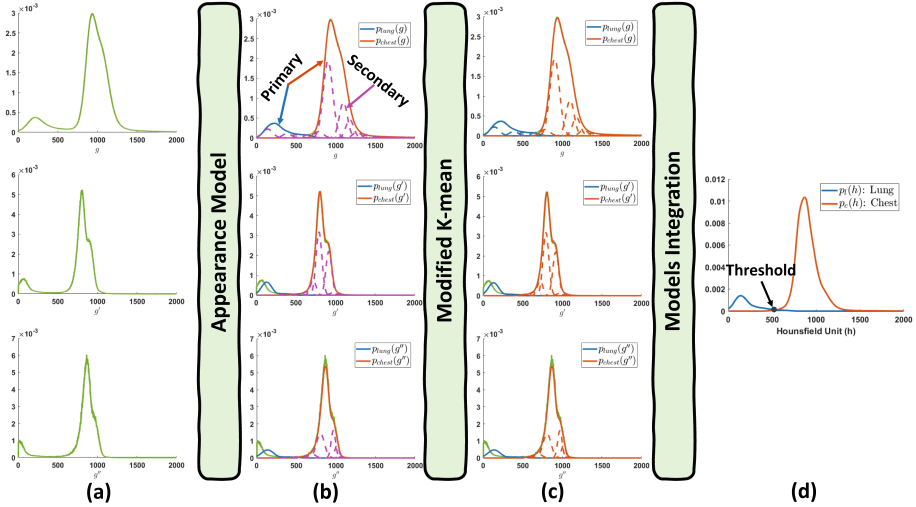


Fig. 2. A visualization of (a) histograms for CT voxel (i.e., g) and its minimum-filtered voxels (i.e., g', g''), (b) their corresponding appearance models, (c) their clustered modes, and (d) final output.

By employing a two-step expectation maximization (EM) algorithm, the mixing weights and Gaussian parameters are computed. The main goal of this algorithm is to maximize the log-likelihood of the empirical data [13]. Afterward, the secondary components are categorized into one of primary components (i.e., lung or chest) using a modified k-means algorithm [27] if the distance between their means and the mean of that primary component is minimized, as illustrated in Fig. 2c. After clustering these secondary components, the appearance probability of a given class (c), as depicted in Fig. 2d, is modeled as follows:

$$p(h|c) = \underbrace{w_c \varphi(h; \theta_c)}_{\text{primary}} + \underbrace{\sum_{\hat{k}} w_{\hat{k}} \varphi(h; \theta_{\hat{k}})}_{\text{secondary}}. \quad (6)$$

2.4 Spatial Model

To investigate the spatial relationship between current region map and its 26 neighbors, a Markov-Gibbs random field (MGRF) approach [1, 13, 15] is

employed to model these spatial relationship. The model adeptly captures pairwise neighboring interactions through analytical estimation of bi-value potentials. These potentials act as distinguishing elements, deciding whether a particular interaction corresponds (i.e., V_{eq}) or deviates (i.e., V_{noneq}) from a specific label. These bi-value potentials can be estimated as follows:

$$V(x, \chi) = \begin{cases} V_{eq} & x = \chi \\ V_{noneq} & x \neq \chi \end{cases} \quad (7)$$

Here, x and χ represent given labels (lung or chest), while V_{eq} and V_{noneq} are calculated as follows:

$$\begin{aligned} V_{eq} &= \frac{X^2}{X-1} \left(f_{eq}(\mathbf{m}) - \frac{1}{X} \right) \\ V_{noneq} &= -V_{eq} \end{aligned} \quad (8)$$

where X and $f_{eq}(\mathbf{m})$ represent the total count of labels and the number of equivalent labels found in all pairwise interactions among its 26-neighbors for a particular region map, respectively.

Finally, the spatial probability for a given offset (ξ, η, ζ) , which defines these neighboring interactions, is calculated based on these estimated potentials, as detailed below:

$$P(\mathbf{m}) = \frac{1}{Z} \exp \left(\sum_{(i,j,z) \in \mathbf{R}} \sum_{(\xi,\eta,\zeta) \in \mathbf{N}} V(m_{i,j,z}, m_{i+\xi,j+\eta,z+\zeta}) \right) \quad (9)$$

Here, Z stands for the normalization factor, whereas N represents the shifts (i.e., distance) for all 26 neighboring elements.

2.5 Optimization Model

To optimize the proposed multi-dimensional joint probability (Eq. 2), a Swin Transformer-based network, specifically Swin-Unet [7], is utilized, as presented in Fig. 1. The strength of the Swin Transformer lies in its ability to efficiently capture long-range dependencies across input data, owing to its hierarchical and non-local attention mechanisms. This capability significantly enhances performance in segmentation tasks. Swin-Unet is built upon an encoder-decoder architecture that incorporates skip connections. These connections facilitate the retention and propagation of low-level features from early layers to deeper layers, thereby aiding in gradient flow, mitigating vanishing gradient issues, and ultimately enhancing the model’s performance. This network utilizes Swin transformer [18], which consists of four layers: linear embedding, encoder, decoder, and linear projection. The linear embedding consists of a convolution layer with a kernel size of 4×4 , followed by a flatten layer and a normalization layer. The encoder layer compresses the input information into a lower-dimensional representation, aiming to identify subtle features, while the decoder layer reconstructs

the output based on this learned representation, with the aim of generating coherent and relevant results. This network utilizes four levels of encoder and decoder layers. Each level of the encoder layer comprises two Swin transformer blocks and patch merging for generating hierarchical features, except for the last level, which consists of two Swin Transformer blocks only. The Swin transformer block, characterized by its hierarchical attention mechanism and shifted window-based self-attention, plays a pivotal role in capturing both local and global features within the input data. This block consists of two consecutive layers, each containing layer normalization (LN) and a multi-head self-attention (MSA) module, followed by a multi-layer perceptron (MLP), as illustrated in Fig. 3a. The structure of this block can be defined as follows:

$$\begin{aligned}
 \hat{o}^l &= W\text{-MSA}(LN(o^{l-1})) + o^{l-1}, \\
 o^l &= MLP(LN(\hat{o}^l)) + \hat{o}^l, \\
 \hat{o}^{l+1} &= SW\text{-MSA}(LN(o^l)) + o^l, \\
 o^{l+1} &= MLP(LN(\hat{o}^{l+1})) + \hat{o}^{l+1}.
 \end{aligned} \tag{10}$$

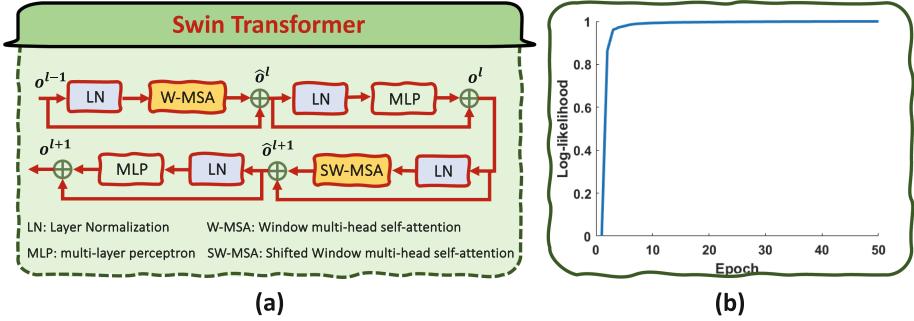


Fig. 3. A schematic illustration of (a) Swin Transformer block, and (b) the normalized log-likelihood.

Here, o^l , and \hat{o}^l represent the output of MLP and (S)W-MSA in the l th layer, respectively. Where, W-MSA self-attention can be defined as follows:

$$Attention(o^l) = \text{Softmax} \left(\frac{QK^T}{\sqrt{d}} + B \right) V \tag{11}$$

where Q , K , and V are learnable matrices with dimension d . Moreover, patch merging layer is responsible for reducing the input twice (i.e., downsampling) by aggregating information from neighboring patches (i.e., four patches), while simultaneously doubling the number of feature maps through the utilization of a linear layer to combine the concatenated features. Furthermore, each level of decoder layer consists of two Swin transformer blocks and patch expanding,

except for the first level, which comprises patch expanding only. Patch expanding layer is responsible for increasing the spatial resolution of feature maps (i.e., upsampling) while integrating contextual information from neighboring patches. Finally, the linear project layer consists of a convolution layer with a kernel size of 1×1 to project the segmentation prediction.

Instead of using this network as a black-box, the Swin-Unet is employed to optimize the proposed joint probability by maximizing the log-likelihood (Eq. 4). It takes input consisting of the joint probability of CT data and its appearance variations. Each probability represented as a separate channel, resulting in a total of six channels, with three channels dedicated to lung features and three channels for chest features. The network’s output serves as potential candidates, aimed at maximizing the log-likelihood of the proposed joint probability. To accomplish this, a spatial probability of these candidates is calculated. Following this, any candidate regions that contribute to an enhancement in the log-likelihood of the joint probability replace the existing region maps. However, this substitution is subject to the condition that the DSC between the old and updated region maps remains above a predefined threshold, ensuring that the updated regions are not chosen arbitrarily. To visually demonstrate the improvement in the log-likelihood of the joint probability, Fig. 3b is provided. This figure illustrates how the log-likelihood of the joint probability steadily rises with each epoch, emphasizing the difference between the two labels. This network is trained in an unsupervised approach, eliminating the requirement for annotation labels (i.e., ground-truth). In this process, if the region maps produced by the network do not enhance the log-likelihood of the proposed joint probability, they are deemed errors. Subsequently, the network autonomously updates itself to rectify these errors, employing a cross-entropy loss function. This function guides the model to reduce the disparity between predicted probabilities (p_l) and the segmentation (t) produced by the proposed joint probability or adjusted by the network. It is applied to both lung and chest probabilities, which can be calculated as follows:

$$l_{CE}(p_l, t) = \underbrace{-t \log(p_l)}_{\text{lung}} - \underbrace{(1-t) \log(1-p_l)}_{\text{chest}} \quad (12)$$

To summarize the steps of the proposed stochastic segmentation system, Algorithm 1 is presented.

3 Experimental Results

The evaluation of the segmentation system is conducted on 28 3D CT chest scans, each displaying different levels of COVID-19 infection severity. The evaluation utilizes an NVIDIA GPU with an Intel Core i9 processor and 64GB of RAM, achieving an average execution time of $256.96_{\pm 62.98}$ s. To assess the effectiveness of our framework compared to methods reliant on training data, an extra set of

Algorithm 1: The core steps of the proposed segmentation system.

input : 3D CT volume.

output: 3D lung segmentation.

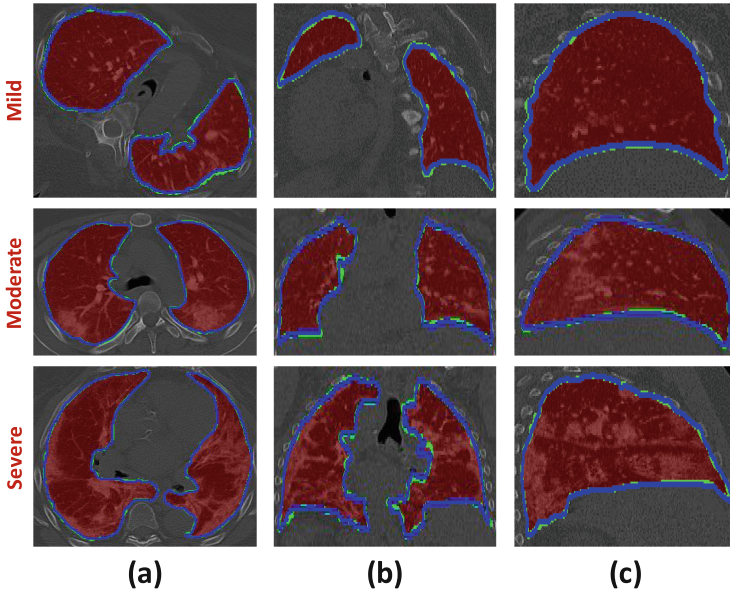
- 1 Use 3D minimum filter to produce various appearance realizations.
 - 2 Estimate the LCG probability for both lung and chest in the CT voxel, as well as two minimum-filtered voxels, using Eq. 6.
 - 3 Use Eq. 9 to compute the MGRF probability for the initial segmentation produced by the model in Step 2.
 - 4 Utilize Eq. 2 to calculate the multi-dimensional joint probability.
- Optimizing log-likelihood:** Iterate through Steps 1–1 until convergence of the log-likelihood of the joint probability is achieved.
- 5 Employ Swin-Unet model to generate a candidate region map.
 - 6 Calculate the MGRF probability of this candidate region map.
 - 7 Substitute the candidate region map with the previous map if its log-likelihood improves and the DSC between the refined and old segmentation remains below a certain threshold.
 - 8 Train the Swin-Unet model using the segmentation, resulted from Step 1.
-

25 3D CT chest scans, comprising a total of 4305 images, was selected for training in our comparative study. The process of gathering information follows the guidelines set forth by the University of Louisville (UofL) Institutional Review Board (IRB), and the study conforms to the ethical standards outlined in the Declaration of Helsinki.

To assess the effectiveness of the proposed segmentation system, its performance is evaluated against its individual components using a diverse array of evaluation metrics, including the overlap coefficient, DSC, absolute volume difference (AVD), and Hausdorff distance (HD) [27], as depicted in Table 1. As illustrated in the table, the appearance model (i.e., LCG) applied to minimum-filtered voxel with a kernel size of 5 yields the worst performance. However, combining the appearance model of original voxel with the minimum-filtered voxels of size 3 and 5 boosts the segmentation performance. Moreover, the integration of the spatial model (MGRF) with the three appearance models using the proposed multi-dimensional joint probability leads to a 4% improvement in the DSC, compared to utilizing the appearance model of minimum-filtered voxel alone. Despite these findings, the proposed segmentation system, optimized using Swin-Unet, surpasses its constituents, achieving a DSC of $95.60_{\pm 1.37}\%$, an overlap coefficient of $91.61_{\pm 2.51}\%$, an AVD of $6.23_{\pm 3.52}$, and an HD of $6.56_{\pm 2.68}$. Furthermore, to visually highlight the promise of the proposed segmentation system, Fig. 4 illustrates various segmentation outcomes corresponding to three levels of respiratory support (i.e., levels 0, 1, and 2) presented in 2D axial, sagittal, and coronal cross-sections. This visualization effectively demonstrates the robustness of the proposed segmentation system. Overall, these outcomes illustrate the precision enhancement achieved by the proposed system, solidifying its superiority in segmentation performance.

Table 1. Qualitative assessment of the proposed segmentation system in comparison to its constituent.

	DSC (%)	Overlap (%)	AVD	HD
<i>LCG</i>	91.03 \pm 5.22	83.92 \pm 8.38	5.57 \pm 6.05	37.09 \pm 15.88
<i>LCG</i> _{Min3}	89.67 \pm 3.58	81.46 \pm 5.74	18.88 \pm 7.79	97.51 \pm 25.89
<i>LCG</i> _{Min5}	82.58 \pm 4.72	70.59 \pm 6.73	41.07 \pm 14.35	101.22 \pm 27.42
<i>LCG</i> _{Combined}	92.60 \pm 3.17	86.38 \pm 5.35	6.33 \pm 4.00	31.91 \pm 17.84
<i>MGRF</i>	93.86 \pm 2.70	88.54 \pm 4.68	6.43 \pm 3.78	30.61 \pm 20.76
<i>Our System</i>	95.60 \pm 1.37	91.61 \pm 2.51	6.23 \pm 3.52	6.56 \pm 2.68

**Fig. 4.** An illustrative example of our system depicted in (a) 2D axial, (b) sagittal, and (c) coronal cross-sections. The red region with a blue border represents the segmentation results, while the green border indicates the ground truth.

Moreover, to underscore the strength and reliability of the proposed system, various state-of-the-art (SOTA) segmentation approaches, including both supervised and unsupervised methods, are employed for comparison. These include 3D Unet [10], DeepLabv3+ [8], Swin-Unet [7], Unet [25], MS-Former [17], and iterative conditional mode (ICM) [6], as summarized in Table 2. As demonstrated in the table, among supervised approaches, DeepLabv3+ exhibits the poorest performance, with a DSC of 83.38 \pm 11.40%, an overlap coefficient of 73.05 \pm 16.50%, an AVD of 38.62 \pm 34.46, and an HD of 50.09 \pm 33.49. Conversely, among unsupervised approaches, MS-Former performs the worst, with a DSC of 76.21 \pm 12.75%, an overlap coefficient of 62.99 \pm 14.47%, an AVD of 39.49 \pm 13.54, and

an HD of $19.03_{\pm 8.01}$. However, the proposed segmentation system surpasses these methodologies, demonstrating its capability in delineating normal and pathological lung areas in CT images. To visually emphasize the potential of the proposed system, Fig. 5 presents various examples of segmentation results across different levels of COVID-19 severity, facilitating a visual comparison between the proposed system and these SOTA methods. As illustrated in the figure, the proposed system demonstrates significant performance compared to these SOTA methods. Moreover, both DeepLabv3+ and MS-Former are unsuitable for segmenting pathological lung regions, as their results are not comparable with those of other SOTA methods. These findings confirm the effectiveness of the proposed system and its potential for accurately outlining both normal and pathological lung regions in CT images.

Table 2. Qualitative assessment of the proposed segmentation system in comparison to existing methodologies.

	DSC (%)	Overlap (%)	AVD	HD
<i>3D Unet</i> [10]	$93.71_{\pm 3.70}$	$88.37_{\pm 6.15}$	$8.66_{\pm 8.91}$	$30.73_{\pm 36.71}$
<i>DeepLabv3+</i> [8]	$83.38_{\pm 11.40}$	$73.05_{\pm 16.50}$	$38.62_{\pm 34.46}$	$50.09_{\pm 33.49}$
<i>Swin-Unet</i> [7]	$92.69_{\pm 6.71}$	$87_{\pm 10.54}$	$10.3_{\pm 10.57}$	$11.81_{\pm 12.32}$
<i>Unet</i> [25]	$94.98_{\pm 1.66}$	$90.49_{\pm 2.99}$	$6.92_{\pm 2.79}$	$13.78_{\pm 21.19}$
<i>MS-Former</i> [17]	$76.21_{\pm 12.75}$	$62.99_{\pm 14.47}$	$39.49_{\pm 13.54}$	$19.03_{\pm 8.01}$
<i>ICM</i> [6]	$94.42_{\pm 2.30}$	$89.52_{\pm 4.04}$	$6.72_{\pm 3.95}$	$26.60_{\pm 20.41}$
<i>Our System</i>	$95.60_{\pm 1.37}$	$91.61_{\pm 2.51}$	$6.23_{\pm 3.52}$	$6.56_{\pm 2.68}$

Furthermore, to demonstrate the robustness of the proposed system, its performance is evaluated on six different lung cancer CT volumes and compared with unsupervised segmentation methods, as outlined in Table 3. As illustrated in the table, the proposed system outperforms these methods, achieving a DSC of $98.37_{\pm 0.42}$, an overlap coefficient of $96.79_{\pm 0.82}$, an AVD of $2.97_{\pm 0.91}$, and an HD of $2.21_{\pm 0.40}$. All these results confirm the superiority of the proposed system in segmenting lung regions.

Table 3. Qualitative assessment of the proposed segmentation system for lung cancer CT volumes compared to existing unsupervised methodologies.

	DSC (%)	Overlap (%)	AVD	HD
<i>MS-Former</i> [17]	$79.05_{\pm 3.83}$	$65.49_{\pm 5.15}$	$36.77_{\pm 5.51}$	$20.32_{\pm 1.87}$
<i>ICM</i> [6]	$97.54_{\pm 0.60}$	$95.21_{\pm 1.14}$	$4.59_{\pm 1.29}$	$31.21_{\pm 23.81}$
<i>Our System</i>	$98.37_{\pm 0.42}$	$96.79_{\pm 0.82}$	$2.97_{\pm 0.91}$	$2.21_{\pm 0.40}$

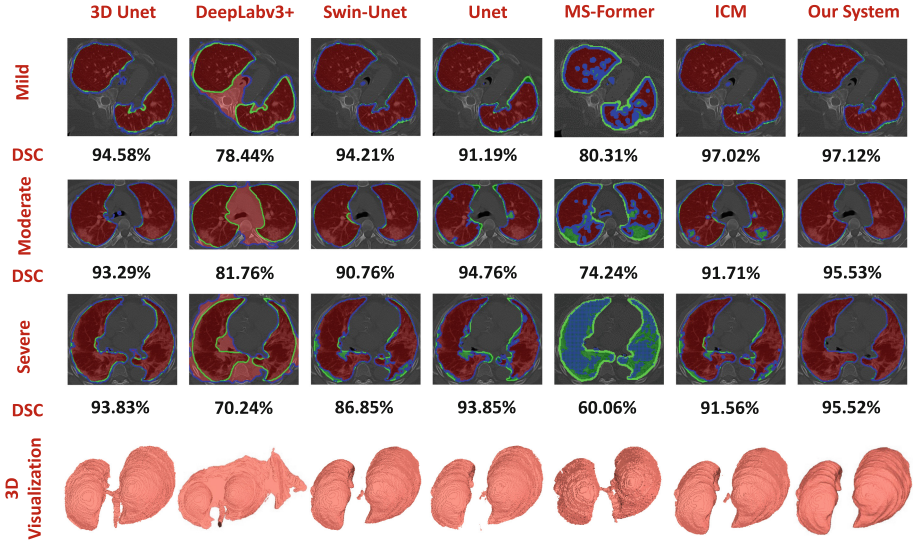


Fig. 5. A visual comparison showcasing the outcomes of our system against those of other existing methods. The red region with a blue border indicates the segmentation results, while the green region with a green border represents the ground truth.

4 Conclusions

This study presented a new segmentation system aimed at precisely outlining lung regions in a 3D CT chest volume. The system employs a stochastic unsupervised probabilistic model, optimized using deep learning model. This probabilistic model integrates an appearance model, created by representing CT radiodensity as a linear combination of Gaussians, with a spatial model generated based on 3D MGRF. This integration is facilitated by introducing a multi-dimensional joint probability that combines the appearance model of various appearances with their respective spatial models. Subsequently, the segmentation produced by this joint probability is further refined using a Swin-Unet transformer, without requiring annotation labels, by updating regions that maximize the log-likelihood of the joint probability. The experimental results showcased a noteworthy enhancement in the system’s accuracy, compared to its individual components, in addition to its superiority over the various SOTA methods. In the future, we aim to investigate the integration of shape-based models into the proposed system and study their impacts on its overall accuracy, as well as validate the proposed system on various pathological lung diseases.

Acknowledgment. This research is supported by Abu Dhabi’s Advanced Technology Research Council via the ASPIRE Award for Research Excellence program.

References

1. Alghamdi, N.S., et al.: Segmentation of infant brain using nonnegative matrix factorization. *Appl. Sci.* **12**(11), 5377 (2022). <https://doi.org/10.3390/app12115377>
2. Alksas, A., et al.: Retinal vascular system segmentation based on non-linear map-based estimation of joint MGRF model. In: 2024 IEEE International Symposium on Biomedical Imaging (ISBI), pp. 1–4. IEEE (2024)
3. Alksas, A., et al.: Advanced octa imaging segmentation: unsupervised, non-linear retinal vessel detection using modified self-organizing maps and joint MGRF modeling. *Comput. Methods Programs Biomed.* **254**, 108309 (2024). <https://doi.org/10.1016/j.cmpb.2024.108309>
4. Balaha, H.M., Balaha, M.H., Ali, H.A.: Hybrid COVID-19 segmentation and recognition framework (HMB-HCF) using deep learning and genetic algorithms. *Artif. Intell. Med.* **119**, 102156 (2021)
5. Batouty, N.M., Saleh, G.A., Sharafeldeen, A., Kandil, H., Mahmoud, A., Shalaby, A., Yaghi, M., Khelifi, A., Ghazal, M., El-Baz, A.: State of the art: lung cancer staging using updated imaging modalities. *Bioengineering* **9**(10), 493 (2022). <https://doi.org/10.3390/bioengineering9100493>
6. Besag, J.: On the statistical analysis of dirty pictures. *J. R. Stat. Soc. Ser. B Stat Methodol.* **48**(3), 259–279 (1986)
7. Cao, H., Wang, Y., Chen, J., Jiang, D., Zhang, X., Tian, Q., Wang, M.: Swin-Unet: Unet-like pure transformer for medical image segmentation. In: Karlinsky, L., Michaeli, T., Nishino, K. (eds.) *Computer Vision – ECCV 2022 Workshops. ECCV 2022. Lecture Notes in Computer Science*, vol. 13803, pp. 205–218. Springer Nature Switzerland (2023). https://doi.org/10.1007/978-3-031-25066-8_9
8. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: *Proceedings of the European Conference on Computer Vision (ECCV)* (2018)
9. Chen, Z., et al.: Deep learning-based bronchial tree-guided semi-automatic segmentation of pulmonary segments in computed tomography images. *Quant. Imaging Med. Surg.* **14**(2), 1636–1651 (2024). <https://doi.org/10.21037/qims-23-1251>
10. Özgün Çiçek, Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3D U-Net: learning dense volumetric segmentation from sparse annotation. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, pp. 424–432. Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-46723-8_49
11. Devi, K.J., Sudha, S.V.: A novel panoptic segmentation model for lung tumor prediction using deep learning approaches. *Soft. Comput.* **28**(3), 2637–2648 (2024). <https://doi.org/10.1007/s00500-023-09569-9>
12. El-Baz, A., et al.: Computer-aided diagnosis systems for lung cancer: challenges and methodologies. *Int. J. Biomed. Imaging* **2013**, 1–46 (2013). <https://doi.org/10.1155/2013/942353>
13. El-Baz, A.S., Gimel'farb, G.L., Suri, J.S.: *Stochastic Modeling for Medical Image Analysis*. CRC Press, Boca Raton (2016), oCLC: 1086143882
14. Fahmy, D., et al.: How AI can help in the diagnostic dilemma of pulmonary nodules. *Cancers* **14**(7), 1840 (2022). <https://doi.org/10.3390/cancers14071840>
15. Farahat, I.S., et al.: An AI-based novel system for predicting respiratory support in COVID-19 patients through CT imaging analysis. *Sci. Rep.* **14**(1), 851 (2024). <https://doi.org/10.1038/s41598-023-51053-9>

16. Gite, S., Mishra, A., Kotecha, K.: Enhanced lung image segmentation using deep learning. *Neural Comput. Appl.* **35**(31), 22839–22853 (2022). <https://doi.org/10.1007/s00521-021-06719-8>
17. Karimijafarbigloo, S., Azad, R., Kazerouni, A., Merhof, D.: Ms-former: multi-scale self-guided transformer for medical image segmentation. In: Oguz, I., et al. (eds.) *Medical Imaging with Deep Learning. Proceedings of Machine Learning Research*, vol. 227, pp. 680–694. PMLR (2024). <https://proceedings.mlr.press/v227/karimijafarbigloo24a.html>
18. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: hierarchical vision transformer using shifted windows. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022 (2021)
19. Liu, C., Pang, M.: Automatic lung segmentation based on image decomposition and wavelet transform. *Biomed. Signal Process. Control* **61**, 102032 (2020). <https://doi.org/10.1016/j.bspc.2020.102032>
20. Missimer, J.H., Emert, F., Lomax, A.J., Weber, D.C.: Automatic lung segmentation of magnetic resonance images: a new approach applied to healthy volunteers undergoing enhanced deep-inspiration-breath-hold for motion-mitigated 4d proton therapy of lung tumors. *Phys. Imaging Radiat. Oncol.* **29**, 100531 (2024). <https://doi.org/10.1016/j.phro.2024.100531>
21. Moosavi, A.S., Mahboobi, A., Arabzadeh, F., Ramezani, N., Moosavi, H.S., Mehrpoor, G.: Segmentation and classification of lungs CT-scan for detecting COVID-19 abnormalities by deep learning technique: U-net model. *J. Family Med. Primary Care* **13**(2), 691–698 (2024). https://doi.org/10.4103/jfmpc.jfmpc_695_23
22. Mousavi Moghaddam, R., Aghazadeh, N.: Lung parenchyma segmentation from CT images with a fully automatic method. *Multimedia Tools Appl.* **83**(5), 14235–14257 (2023). <https://doi.org/10.1007/s11042-023-16040-2>
23. Murugappan, M., Bourisly, A.K., Prakash, N.B., Sumithra, M.G., Acharya, U.R.: Automated semantic lung segmentation in chest CT images using deep neural network. *Neural Comput. Appl.* **35**(21), 15343–15364 (2023). <https://doi.org/10.1007/s00521-023-08407-1>
24. Osadebey, M., Andersen, H.K., Waaler, D., Fossaa, K., Martinsen, A.C.T., Pedersen, M.: Three-stage segmentation of lung region from CT images using deep neural networks. *BMC Med. Imaging* **21**(1), 1–9 (2021). <https://doi.org/10.1186/s12880-021-00640-1>
25. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W., Frangi, A. (eds.) *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015. MICCAI 2015, Lecture Notes in Computer Science*, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
26. Sharafeldeen, A., et al.: Accurate segmentation for pathological lung based on integration of 3D appearance and surface models. In: *2023 IEEE International Conference on Image Processing (ICIP). IEEE* (2023). <https://doi.org/10.1109/icip49359.2023.10222525>
27. Sharafeldeen, A., Elsharkawy, M., Alghamdi, N.S., Soliman, A., El-Baz, A.: Precise segmentation of COVID-19 infected lung from CT images based on adaptive first-order appearance model with morphological/anatomical constraints. *Sensors* **21**(16), 5482 (2021). <https://doi.org/10.3390/s21165482>
28. Sharafeldeen, A., Khelifi, A., Ghazal, M., Yaghi, M., Contractor, S., El-Baz, A.: Automated segmentation of lung regions in 3D CT scans using hybrid

- unsupervised-supervised models. In: 2024 IEEE International Conference on Image Processing (ICIP). IEEE (2024)
29. Shi, C., Cheng, Y., Wang, J., Wang, Y., Mori, K., Tamura, S.: Low-rank and sparse decomposition based shape model and probabilistic atlas for automatic pathological organ segmentation. *Med. Image Anal.* **38**, 30–49 (2017). <https://doi.org/10.1016/j.media.2017.02.008>
 30. Stover, D.E., White, D.A., Romano, P.A., Gellene, R.A., Robeson, W.A.: Spectrum of pulmonary diseases associated with the acquired immune deficiency syndrome. *Am. J. Med.* **78**(3), 429–437 (1985). [https://doi.org/10.1016/0002-9343\(85\)90334-1](https://doi.org/10.1016/0002-9343(85)90334-1)



Deep Spherical Superpixels

Rémi Giraud¹✉ and Michaël Clément²

¹ Univ. Bordeaux, Bordeaux INP, IMS, CNRS UMR 5218, Bordeaux, France
remi.giraud@ims-bordeaux.fr

² Univ. Bordeaux, Bordeaux INP, LaBRI, CNRS UMR 5800, Bordeaux, France
michael.clement@labri.fr

Abstract. Over the years, the use of superpixel segmentation has become very popular in various applications, serving as a preprocessing step to reduce data size by adapting to the content of the image, regardless of its semantic content. While the superpixel segmentation of standard planar images, captured with a 90° field of view, has been extensively studied, there has been limited focus on dedicated methods to omnidirectional or spherical images, captured with a 360° field of view. In this study, we introduce the first deep learning-based superpixel segmentation approach tailored for omnidirectional images called DSS (for Deep Spherical Superpixels). Our methodology leverages on spherical CNN architectures and the differentiable K -means clustering paradigm for superpixels, to generate superpixels that follow the spherical geometry. Additionally, we propose to use data augmentation techniques specifically designed for 360° images, enabling our model to efficiently learn from a limited set of annotated omnidirectional data. Our extensive validation across two datasets demonstrates that taking into account the inherent circular geometry of such images into our framework improves the segmentation performance over traditional and deep learning-based superpixel methods. Our code is available online (<https://github.com/rgiraud/dss>).

Keywords: Superpixels · Omnidirectional Images · Spherical CNN

1 Introduction

The vast majority of computer vision methods are tailored for standard RGB images, *i.e.*, captured with a standard 90° field of view (FoV). However, acquisition devices with wider FoV have become more and more popular in the recent years. In particular, omnidirectional images with a $360^\circ \times 180^\circ$ FoV are very interesting to capture the entire environment of a scene. Over the literature, such imagery may be equally referred as omnidirectional, spherical, 360° , or even panoramic. Naturally, such acquisition introduces distortions when projecting the capture on a planar 2D image. Nevertheless, many dedicated methods have been successfully applied on

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78347-0_5.

these images, for example for scene reconstruction [21], semantic segmentation [27] for autonomous driving, or in the context of mixed or virtual reality [18].

To efficiently apply deep learning-based architectures to these images, a few adjustments must be made to consider their specific geometry. For instance, the input images are horizontally circular so the pixels of the first column should be considered spatially adjacent to the pixels of the last column. Some methods explicitly take into account these geometrical properties, for instance with spherical convolutional neural networks (SCNNs) that have demonstrated higher performance on 360° images than standard CNNs [6]. Nevertheless, as for any deep learning-based method, a significant amount of annotated data is necessary for an efficient training, especially when tackling segmentation applications.

For regular standard images, various segmentation datasets are available with different content, resolution or precision in the annotations. However, only a few spherical image datasets are available, such as SUN360 [25] or Matterport3D [4]. Moreover, due to the tediousness of a pixel-wise semantic segmentation process, they generally only provide layout, depth or camera pose information [19]. In the context of autonomous driving, many datasets contain pixel-wise semantic annotations but the FoV is generally limited to standard rectangular acquisition [7, 8], or the images are captured by a fisheye lense introducing other distortions [29]. Hence, deep learning segmentation methods that are applied to 360° imagery may highly necessitate specific data augmentation strategies [21, 27].

In a more general context of image segmentation methods, non-semantic decompositions into superpixels offer numerous benefits. These methods regularly group pixels into homogeneous and connected regions, respecting the image contours. They have mainly been popularized by SLIC [1], a simple method that uses a locally constrained iterative K -means clustering, computed on color and position features. Then, many derived methods have been proposed, such as the non-iterative SNIC method [2], LSC [5] which expands the feature space of SLIC, or SCALP [11] that computes a color consistency along the path between a pixel and the centroid of its superpixel. Other methods like GMMSP [3] propose different strategies, such as using a Gaussian Mixture Model.

The first superpixel method tailored for spherical images was proposed in [22], extending SLIC. The spherical geometry is considered in the clustering distance, that is computed using the 3D positions of pixels on the sphere. The produced superpixels are regular on the 3D sphere domain and are able adapt to the distortions of objects induced by the projection on the 2D planar image, leading to higher segmentation performance compared to planar methods. Following, many planar superpixel algorithms have had their omnidirectional counterparts, such as SSNIC [20], SphLSC and SphSPS (or SphSCALP) [9].

Nevertheless, over the years, all these traditional approaches have started to report saturated performance over the segmentation benchmarks. With the Superpixel Sampling Network (SSN) method [12], a first deep learning framework has been proposed to compute a segmentation into superpixels. SSN and following methods, *e.g.*, [26], enable to improve the segmentation accuracy by computing more advanced features, with the use of a CNN trained on higher-level annotated segmentations (for example from semantic segmentations). However, these deep learning methods have only been designed for standard planar images.

Contributions. In this work, we propose the first deep learning-based method called Deep Spherical Superpixels (DSS), able to segment omnidirectional images into spherical superpixels. The contributions of this work are listed as follows:

- i We introduce the first deep learning-based superpixel segmentation method tailored for omnidirectional images, leveraging spherical CNN architectures and the differentiable K -means superpixel algorithm;
- ii We make use of specific data augmentation strategies designed for 360° images, whose effectiveness is demonstrated through an ablation study;
- iii We comprehensively evaluate the proposed method against state-of-the-art approaches, including both traditional planar and spherical approaches as well as deep learning-based methods, evaluated for the first time on the spherical superpixel segmentation task;
- iv We propose a quantitative validation on the Panorama Segmentation Dataset (PSD) [22], the reference for spherical superpixels, on initial and noisy images, and also on a newly considered omnidirectional road dataset, Wild PANoramic Semantic Segmentation (WildPASS) [28];
- v The source code of our method is made available to the research community.

2 Deep Spherical Superpixels Method

In this Section, we introduce our proposed Deep Spherical Superpixels (DSS) method. First, we present the Superpixel Sampling Network (SNN) [12] framework that we use as basis for our method (Sect. 2.1). Then, we detail the 360° coordinates system (Sect. 2.2) and our modifications of SSN to generate spherical superpixels (Sect. 2.3). Finally, we present the 360° -specific data augmentation used to enable our model to efficiently learn from a limited set of annotated omnidirectional data (Sect. 2.4).

2.1 Superpixel Sampling Network

In the superpixel segmentation literature, the Simple Linear Iterative Clustering (SLIC) algorithm is one the most simple yet accurate method [1]. It performs a locally constrained K -means clustering starting from a regular sampling grid. This clustering relies on a spatial and a color distance between each pixel and a superpixel centroid. Although SLIC is interesting for its rapidity and ease to use, its clustering accuracy can be limited since it is only based on RGB or Lab image features.

In [12], an end-to-end framework is proposed using a convolutional neural network (CNN) trained to learn how to provide more advanced features as input to a differentiable SLIC clustering algorithm. The network is trained to produce superpixels that are contained into higher-level annotated segmentations (for example from semantic segmentations). In particular, the integration of SLIC into a deep learning framework is possible in a differentiable manner by considering *soft* mappings of pixels to superpixels. At inference time, the final *hard*

mapping, associating a pixel to a unique superpixel, is only computed to generate the final segmentation.

The SSN model takes as input images of size $N = h \times w$, represented with 5 channels corresponding to *Lab* color features (3 channels) and *xy* pixel coordinates (2 channels). The goal of the model is to learn deep features that are more suitable to perform a differentiable clustering into superpixels. To achieve this, the SSN model uses a CNN composed of three blocks, each with two convolutional layers, batch normalization and ReLU activation, with a max pooling layer applied after each block. For the output, feature maps of each block are upsampled to the original image size (if necessary, for the second and third blocks) and concatenated. The original *Lab* and *xy* features are also concatenated into the output feature maps, resulting in D -dimensional pixel features (*i.e.*, 5 channels from input features and $D - 5$ learned deep features). In practice, the SSN model used $D = 20$ in their experiments. For more details about this architecture, the reader can refer to [12].

These learned features are then fed to the aforementioned differentiable clustering to compute soft assignments of pixels to superpixels. These soft assignments are in turn used to compute a loss function tailored for the desired superpixel properties. For example, to obtain superpixels matching semantic segmented objects, the loss is comprised of two terms: (i) a pixel-wise cross-entropy term between ground-truth semantic segmentation and predicted soft superpixels and (ii) a compactness term which encourages superpixels to have low spatial variance.

The method is therefore end-to-end trainable and can learn deep pixel features tailored for subsequent superpixels properties. In the following, we present how to adapt this approach to the specific case of generating spherical superpixels for omnidirectional images.

2.2 Spherical Geometry

The projection system between the planar equirectangular 2D space and the 3D spherical space is depicted in Fig. 1. This relationship can be understood through the projection of vertical and horizontal coordinates of the plane onto the sphere’s meridians and latitude circles. This process creates a spherical image where the width w is double the height h . It implies a horizontal continuity in the planar image domain that characterizes omnidirectional images. Hence, each pixel $p = [j, i]$ in the 2D space matches a 3D point $X = [x, y, z]$ on the unit sphere following the equations:

$$p = \begin{bmatrix} j = \lfloor \frac{\theta w}{2\pi} \rfloor \\ i = \lfloor \frac{\phi h}{\pi} \rfloor \end{bmatrix} \leftrightarrow X = \begin{bmatrix} x = \sin(\frac{y\pi}{h})\cos(\frac{2x\pi}{w}) \\ y = \sin(\frac{y\pi}{h})\sin(\frac{2x\pi}{w}) \\ z = \cos(\frac{y\pi}{h}) \end{bmatrix}, \quad (1)$$

where $\theta = \arctan2(y, x)$ is the azimuthal angle, and $\phi = \arccos(z)$ is the polar angle. Note that this mapping of coordinates considers that $j \in [-\frac{w}{2}, \frac{w}{2}]$ so to map x to $[0, w]$, we have $x \leftarrow x + w$ when $x \leq 0$.

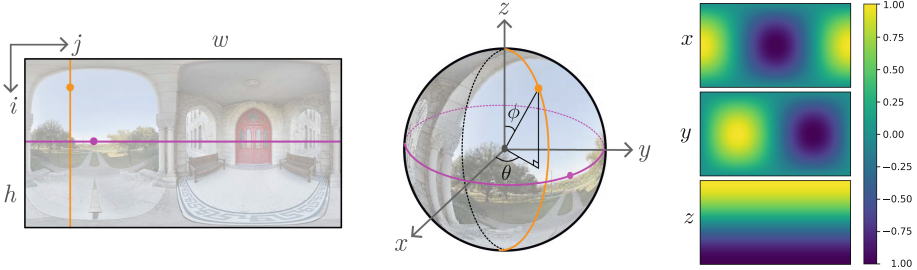


Fig. 1. 2D Planar and 3D spherical system coordinates. A pixel at position $[j, i]$ in the 2D space is mapped to a 3D point $[x, y, z]$ on the unit sphere following (1). This point can also be represented by its respective azimuthal and polar angles θ and ϕ .

2.3 Spherical Superpixel Clustering Network

In this Section, we describe our adaptation of the K -means differentiable superpixel clustering network [12] to provide superpixels that are regular over the spherical domain. We use the same CNN architecture as basis for our method.

Features and Superpixels Initialization. As input for the CNN, we use the *Lab* color features of the $N = h \times w$ pixels, denoted as $F_c \in \mathbb{R}^{N \times 3}$. The pixel coordinates are also given as input, but instead of the 2D pixel positions, we provide the 3D spherical coordinates $F_s \in [-1, 1]^{N \times 3}$. To match the coordinates domain, we normalize the *Lab* features F_c to also lie in $[-1, 1]$.

With classical 2D images, superpixel clusters are usually initialized by a regular sampling on the 2D grid. However, this strategy is not ideal with omnidirectional images as it does not respect the underlying 3D geometry. To overcome this issue, many spherical sampling strategies have been compared for superpixel clusters initialization [9, 20]. In our proposed DSS method, as in [9], we use Hammersley sampling [24] to rapidly provide an appropriate set of K 3D points that are uniformly distributed on the unit sphere (see Fig. 2(a)). From this set of 3D points, we define an initial label map by a nearest neighbor computation on the 3D pixel position X (see Fig. 2(b)). Such spherically uniform sampling implies a sparser 2D sampling on the planar image near vertical borders. Classical planar methods that consider an initial regular grid would produce very irregular oversegmentation around the sphere’s poles, as shown later in Sect. 3.3. From this label map, we extract the initial superpixel features with an average pooling.

Neighborhood-Based Distance. In the original SLIC method [1], the K -means clustering is locally constrained so each superpixel can only aggregate a pixel in a fixed sized square window centered on the superpixel barycenter. For efficient implementation purposes, the K -means-based differentiable clustering of SSN [12] slightly differs by iteratively computing the pixel association within the 9-th superpixel neighborhood of the initialization map. Therefore, the core of the clustering distance computation is geometry-agnostic, once the superpixel neighbors are identified. In our context, we can compute for each superpixel a

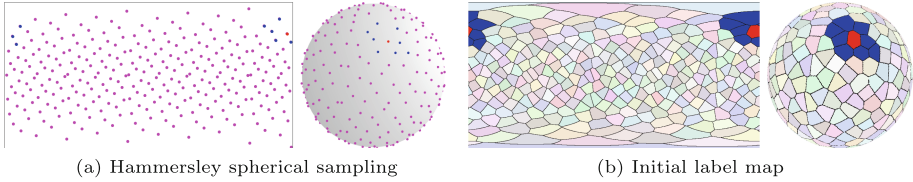


Fig. 2. Spherical label map initialization. (a) A Hammersley sampling with $K = 300$ centroids points is computed on the unit sphere. Note the lower sampling density at the vertical borders, corresponding to the sphere’s poles. (b) Corresponding label map, where each pixel is associated to the closest Hammersley barycenters, producing regular regions on the sphere. The 8 neighbors of the red superpixel (closest in the spherical space) are represented in blue. (Color figure online)

n -th neighborhood with a nearest neighbors distance on their 3D barycenters in the spherical space. Such neighborhood is represented in Fig. 2.

Therefore, contrary to the planar square sampling, our method can define without ambiguity a $n \in \llbracket 0, N \rrbracket$ -th neighborhood. In practice, we use a $n = 9$ neighborhood, as in SSN.

Horizontally Circular Clustering. 360° images are particularly characterized by their horizontally circular nature. This aspect is not considered in standard CNNs, which typically use zero padding strategies for convolutions and where the final receptive field may be also lower than the image dimension. In the context of spherical superpixel clustering, without any semantic aggregation of clustered regions, using standard convolutions is highly irrelevant since we would observe a discontinuity in the segmentation at the image borders.

For example, Fig. 3(a) shows the result obtained by using a standard zero padding strategy in the CNN layers. With 3×3 convolution kernels, the features extracted for pixels at $j = 0$ and $j = w - 1$ are not consistent with the ones of their neighborhood, which disrupts the selection of their closest superpixel among the 9 closest. When computing the hard clustering association, border pixels are generally associated to a disconnected region resulting in the appearance of an artificial vertical border in the spherical space, as for planar methods.

To take into account this horizontally circular geometry into our model, we propose to use a *spherical CNN* with a more natural circular padding strategy, as in [16, 23]. Our spherical CNN uses a horizontal circular (or periodic) padding of half size of the kernel at each step requiring padding (convolutional or max pooling layers). A replicate padding strategy is used for vertical padding. Hence, the spherical CNN is fully able to preserve the 360° geometry in the final clustering and to compute relevant features at the borders. Note that other strategies may be possible, such as applying a large input circular padding as a preprocessing [17], but with many successive convolutions, this leads to handle significantly larger images, and thus to higher memory and time consumption.

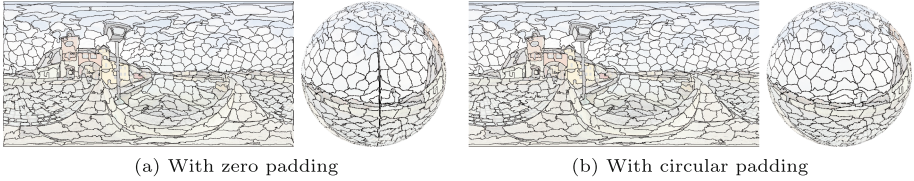


Fig. 3. Impact of the circular padding on the superpixel segmentation. (a) With standard zero padding, the CNN features of pixels at $j = 0$ and $j = w - 1$ are not consistent with neighborhood, leading to a vertical border in the spherical space, as for planar methods. (b) With circular padding, the features remain consistent on the borders and the method is able to fully consider the geometry of the omnidirectional images.

Loss Function. Deep pixel features from our spherical CNN are fed to the differentiable clustering method to produce soft assignments $\mathcal{S}_{\text{soft}}$ of spherical superpixels. As in SSN, the model is trained with a loss comprised of a pixel-wise cross-entropy with ground-truth segmentation \mathcal{G} denoted L_{seg} , and a compactness term L_{compact} to enforce superpixels with low spatial variance:

$$L = L_{\text{seg}}(\mathcal{G}, \mathcal{S}_{\text{soft}}) + \lambda L_{\text{compact}}(F_s, \mathcal{S}_{\text{soft}}). \quad (2)$$

Region Connectivity. After training, to compute the final superpixel segmentation of an image, a last step ensures the connectivity of the produced regions as for most superpixel clustering methods [1, 12]. This is simply done by aggregating the smallest disconnected regions to the largest and nearest one but taking into account the circular aspect.

2.4 360°-Specific Data Augmentation

In the context of 360° imagery, the lack of extensive image datasets with segmentations makes it hard to train neural networks efficiently. To mitigate this data limitation, the use of data augmentation strategies is crucial. While simple augmentation techniques such as flips, blurs, and noise addition are applicable, they may be insufficient to provide enough diversity to the training process. However, many other conventional data augmentation strategies may alter the intrinsic 360° geometry and should not be used for such images. For instance, rotations or crops, as used in SSN [12], compromise the spherical geometry, leading to the loss of the horizontal mirror effect and the spatial distortion of the 2D label map. Using such augmentation techniques would lead the model to learn to provide irregular superpixels in the spherical space with artificial vertical borders at the edges, as for planar methods (see Fig. 3(a)).

To overcome these challenges, we propose to use data augmentation techniques tailored explicitly for 360° images. A straightforward augmentation technique would consist in horizontally rolling the 360° image and its ground-truth [14]. As stated in [16], such data augmentation strategy does not bring any diversity in a pure CNN network. Nevertheless, in our context, since average

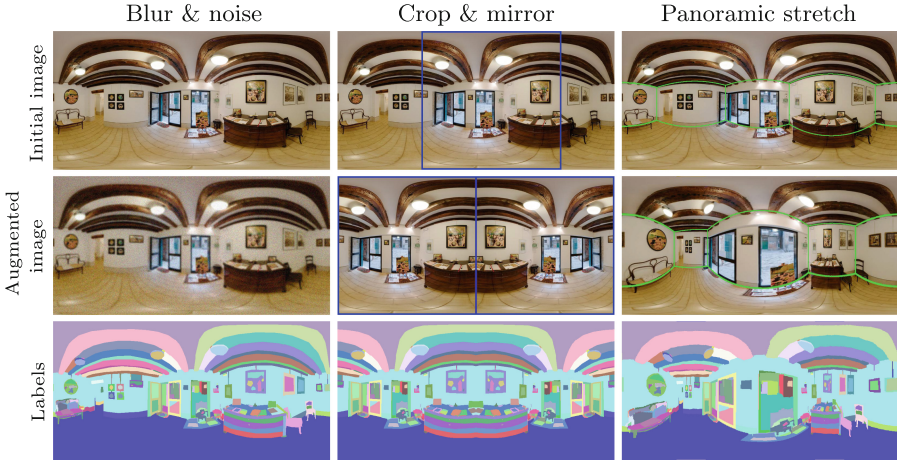


Fig. 4. Example of data augmentation used during training. **Left:** Standard Gaussian blur and noise (here with respective maximal variance $\sigma = 20$ and $\sigma = 2$). The ground-truth labels are not impacted. **Middle:** Crop & mirror strategy. A random crop of half-width is selected (represented by the green square) and mirrored to form a new 360° image. This method combines horizontal rolling, flipping and also creates information at the mirror border. **Right:** Panoramic stretch [21] to introduce distortions in the 360° image (here with parameters $k_x = 0.5$, $k_y = 1.25$ that correspond to a respective enlargement and a shrinking of the areas where $|x| \approx 1$ and $|y| \approx 1$). The layout of the scene is represented by the green lines to more easily apprehend the distortion.

superpixel features are extracted according to an spherical initialization label map, a roll of the image may have a different impact on the produced segmentation. To go further, we also propose to combine random half-width cropping and horizontal mirroring of the input image and ground-truth (see Fig. 4(middle)). This way, in a single transformation, we combine rolling and flipping while creating information at the mirror border.

Finally, we use the *panoramic stretch* approach of [21] to introduce spatial distortions. To stretch a 360° image, $[x, y, z]$ coordinates are simply multiplied by a respective factor $[k_x, k_y, k_z]$ and projected back to the sphere. Pixel values are then computed using bilinear interpolation. Since setting k_z would affect the projection of x and y values the same way, authors propose to only set k_x and k_y parameters. The 3D coordinates maps in Fig. 1 represent the image area that would be affected by increasing one of the parameters. For instance, setting $k_y < 1$ would zoom on the region where y values are close to -1 and 1 (see Fig. 4(right)). We refer the reader to [21] to more details on the stretching algorithm and to our supp. mat. for additional examples.

With such data augmentation, we are able to greatly enrich the training dataset while preserving the spherical geometry of 360° images. We demonstrate the improvement of performance obtained using these techniques during training in Sect. 3.2.

3 Results

3.1 Validation Framework

Datasets. In our experiments, we considered two relevant spherical segmentation datasets containing various accurately segmented objects (see examples in supp. mat.). The first dataset called Panorama Segmentation Dataset (PSD) [22] is the reference one and contains 75 images of 512×1024 pixels from the SUN360 dataset [25]. The ground-truth manual segmentations from [22], contain an average number of 510 objects with an average size of 1334 pixels. To fairly compare deep learning methods, we respectively consider 55, 5 and 15 images for the train, validation and test sets. In Sect. 3.3, we also compare the performance on PSD images affected by an additive white Gaussian noise of variance 20.

To further demonstrate the performance of DSS, we choose to consider for the first time in spherical superpixel methods evaluation, the Wild PANoramic Semantic Segmentation (WildPASS) dataset [28], containing 500 omnidirectional natural road images. We resize the images to 512×1024 and split the dataset into respectively 300, 100 and 100 images for train, validation and test sets.

Parameter Settings. Our data augmentation is applied on-the-fly during training. It includes (i) applying a random Gaussian blur with a variance $\sigma \in [0, 2]$, (ii) adding Gaussian noise of variance $\sigma \in [0, 20]$, (iii) random flipping, horizontal rolling and half-width random crop and mirror with a 0.5 probability, and (iv) panoramic stretching with random parameters k_x and k_y between 0.5 and 2. During training, $\lambda = 1$ in (2) and images are downsized to 256×512 pixels, so our model can understand the whole scene’s geometry, contrary to the 201×201 crops used in [12]. We refer the reader to the supp. mat. for training details.

Evaluation Metrics. The main challenge in superpixel segmentation is the ability to produce superpixels that are contained into the image objects, with respect to a ground-truth segmentation. Regularity is also an important aspect to interactive applications or to later extract significant neighborhoods [10]. Since these criteria are generally contradictory, efficiently maximizing both is usually the bottleneck of superpixel methods. These aspects can be relevantly evaluated with state-of-the-art dedicated metrics [10]. In the following, we denote superpixel segmentation as $\mathcal{S} = \{S_i\}$ and ground-truth segmentation as $\mathcal{G} = \{G_j\}$ with their respective borders $\mathcal{B}(\mathcal{S})$ and $\mathcal{B}(\mathcal{G})$.

The mainly reported measure is the segmentation accuracy, with the Achievable Segmentation Accuracy (ASA) [13] such that:

$$\text{ASA}(\mathcal{S}, \mathcal{G}) = \frac{1}{\sum_{S_i \in \mathcal{S}} |S_i|} \sum_{S_i} \max_{G_j \in \mathcal{G}} |S_i \cap G_j|. \quad (3)$$

This aspect can also be evaluated by focusing on the contour adherence of superpixels to the object borders, using the Boundary-Recall (BR) such that:

$$\text{BR}(\mathcal{S}, \mathcal{G}) = \frac{1}{|\mathcal{B}(\mathcal{G})|} \sum_{p \in \mathcal{B}(\mathcal{G})} \delta[\min_{q \in \mathcal{B}(\mathcal{S})} \|p - q\| < \epsilon], \quad (4)$$

Table 1. Ablation study of the proposed DSS method on PSD and noisy PSD images on ASA (\uparrow), CD/BR (\downarrow) and GGR (\uparrow). CD is given for BR = 0.8. Best and second best results are respectively in bold and underlined font.

Data augmentation				PSD			Noisy PSD		
Gaussian blur&noise	Horizontal crop&mirror	Panoramic stretch	Circular padding	ASA	CD/BR	GGR	ASA	CD/BR	GGR
-	-	-	✓	0.862	0.134	0.385	0.858	0.139	0.386
✓	-	-	✓	0.877	<u>0.119</u>	0.444	0.868	0.132	0.461
✓	✓	-	✓	<u>0.888</u>	0.117	<u>0.413</u>	0.883	0.124	<u>0.423</u>
✓	✓	✓	-	0.887	0.124	0.387	<u>0.884</u>	0.134	0.390
✓	✓	✓	✓	0.890	0.122	0.388	0.886	<u>0.132</u>	0.392

with ϵ a distance threshold set to 2 pixels [10], and $\delta[a] = 1$ when a is true and 0 otherwise. Since it only measures recall, BR should be compared to the Contour Density (CD), *i.e.*, the proportion of border pixels of the generated superpixels.

Finally, to evaluate the regularity aspect, we use the Generalized Global Regularity (GGR) metric that adapts the metric proposed in [10] to 360° images [9]. This metric evaluates the convexity, balanced pixel distribution, contour smoothness of each shape and also how homogeneous the shape distribution is within the segmentation. We refer the reader to [9] to more details on the GGR metric.

3.2 Ablation Study

In Table 1, we report the impact of each data augmentation strategy and the spherical CNN architecture, *i.e.*, using circular padding instead of zero padding [12] on the PSD and noisy PSD images for an average number of $K = 500$ superpixels. Each augmentation strategy increases the training efficiency in terms of segmentation accuracy, while the circular padding logically improves the spherical regularity by cancelling the artificial horizontal border of the segmentation. This confirms the interest of improving the original SSN method with spherical CNN architecture and specific augmentation strategies for 360° images.

3.3 Comparison to State-of-the-Art Methods

Compared Methods. In our experiments, we compare DSS to the spherical methods: SSLIC [30], SSNIC [20], SphLSC and SphSPS [9]. We also compare to some recent planar methods: LSC [5], SNIC [2], GMMSP [3], and SSN [12]. All methods are used with the default regularity parameters. For the SSLIC method [30], that does not have one, we use a color weight of 20 to try to optimize its segmentation accuracy. For SSN [12], we compare to both the initial network trained on the BSD dataset [15] containing planar natural images (SSN-BSD) and to a retrained network on the targeted dataset (SSN-PSD, SSN-WP).

Evaluation of Performance. We compare the proposed DSS to the spherical methods in terms of segmentation accuracy (ASA) and also contour adherence (CD/BR) for several superpixel numbers required K , on the PSD images Fig. 5(a), noisy PSD images Fig. 5(b) and WP images Fig. 5(c).

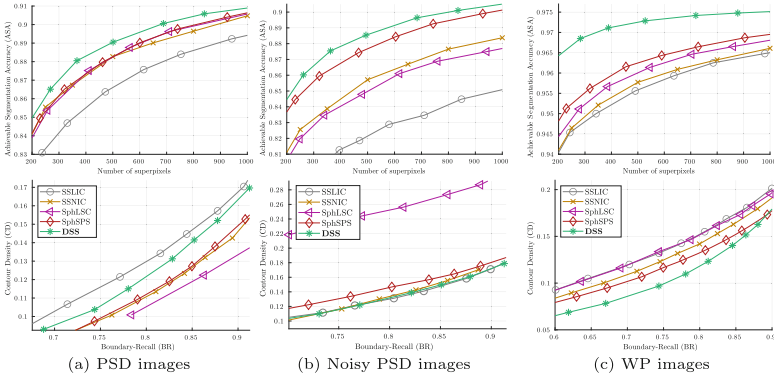


Fig. 5. Comparison of DSS to state-of-the-art methods. **Top:** Segmentation accuracy evaluated with ASA (3). **Bottom:** Contour adherence in terms of CD vs BR (4).

We observe that DSS obtains the highest segmentation accuracy (ASA) on all type of images. We can also see that our method is robust to noise contrary to most state-of-the-art methods that present a significant loss of performance on such slightly altered images. Finally, we can note that DSS superpixels also have the highest contour adherence (lowest CD/BR) compared to other methods, only except on noise-free PSD images. This can be simply explained by the fact that our method, as SSN, does not explicitly integrate a contour adherence loss and that the ground-truth segmentations in the PSD dataset contain many annotations of very thin objects that impact such metric.

In Table 2, we report results for $K = 500$, also including the regularity metric (GGR), and performance obtained with planar methods. We observe that GGR discriminates well the planar and spherical methods. DSS is among the spherical methods, having higher spherical regularity than planar methods, and it also preserves its regularity in the presence of noise.

Table 2. Quantitative comparison of DSS to state-of-the-art methods for an average number of $K = 500$ superpixels on ASA (\uparrow), CD/BR (\downarrow) and GGR (\uparrow). CD is given for BR=0.8. Best and second best results are respectively in bold and underlined font.

	PSD			Noisy PSD			WP			
	ASA	CD/BR	GGR	ASA	CD/BR	GGR	ASA	CD/BR	GGR	
Planar	LSC [5]	0.877	0.138	0.347	0.844	0.303	0.334	0.962	0.153	0.313
	SNIC [2]	0.864	0.129	0.361	0.852	0.139	0.357	0.958	0.146	0.322
	GMMSP [3]	0.877	0.136	0.339	0.849	0.329	0.328	0.963	0.157	0.306
	SSN-BSD [12]	0.879	0.119	0.328	0.863	0.147	0.321	0.967	0.134	0.296
	SSN-PSD/WP [12]	0.887	0.114	0.334	0.873	0.141	0.328	<u>0.972</u>	<u>0.120</u>	0.303
Spherical	SSLIC [30]	0.866	0.130	0.421	0.821	0.130	0.383	0.956	0.152	0.399
	SSNIC [20]	0.883	<u>0.110</u>	0.462	0.857	0.134	0.399	0.958	0.142	<u>0.410</u>
	SphLSC [9]	0.882	0.105	0.397	0.850	0.252	0.357	0.960	0.152	0.360
	SphSPS [9]	0.883	0.112	<u>0.452</u>	<u>0.877</u>	0.146	0.389	0.962	0.133	0.411
	DSS	0.890	0.122	0.388	0.886	<u>0.132</u>	<u>0.392</u>	0.973	0.118	0.356

Compared to SSN, we can first notice that SSN trained on the BSD does not generalize very well when applied on PSD or WP images. It demonstrates the capacity of CNNs to extract semantic information and that performance of generalization may highly depend on the similarity of annotations. We also observe that DSS slightly outperforms SSN retrained on the PSD and WP datasets, in terms of segmentation accuracy. SSN is able to train its network by providing image crops, which is a much more efficient learning strategy than to provide the whole image, as we have to do in DSS. Nevertheless, with our data augmentation strategy, we can maintain the same level of accuracy while generating spherical superpixels that may follow the deformed objects.

Finally, qualitative results are respectively shown on PSD, noisy PSD and WP images in Figs. 6, 7, 8. For planar methods, we can note the projection irreg-



Fig. 6. Qualitative comparison on PSD images, for planar (left) and spherical methods (right) for two superpixel numbers $K = 1200$ (top-left) and $K = 400$ (bottom right).

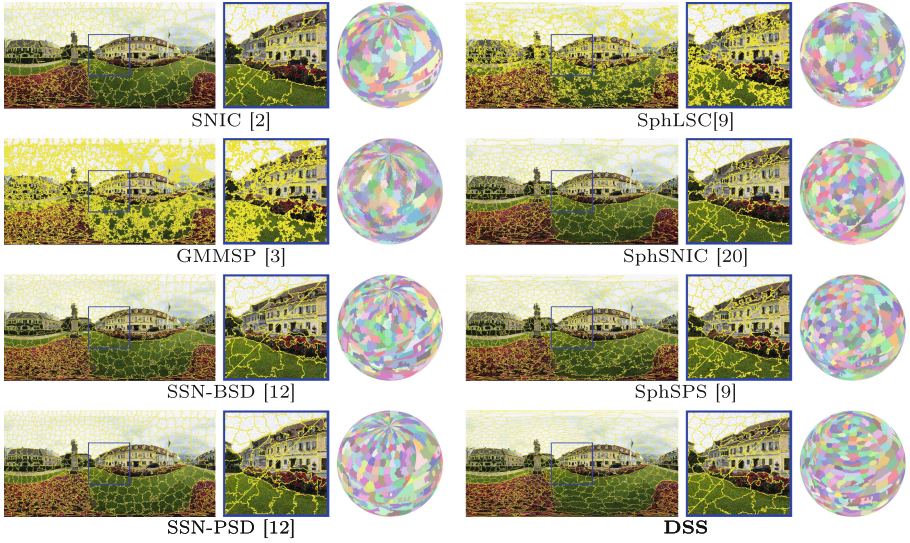


Fig. 7. Qualitative comparison on a noisy PSD image for planar (left) and spherical methods (right) for two superpixel numbers $K = 1200$ (top-left) and $K = 400$ (bottom right). DSS is able to preserve its regularity and accuracy compared to most methods.

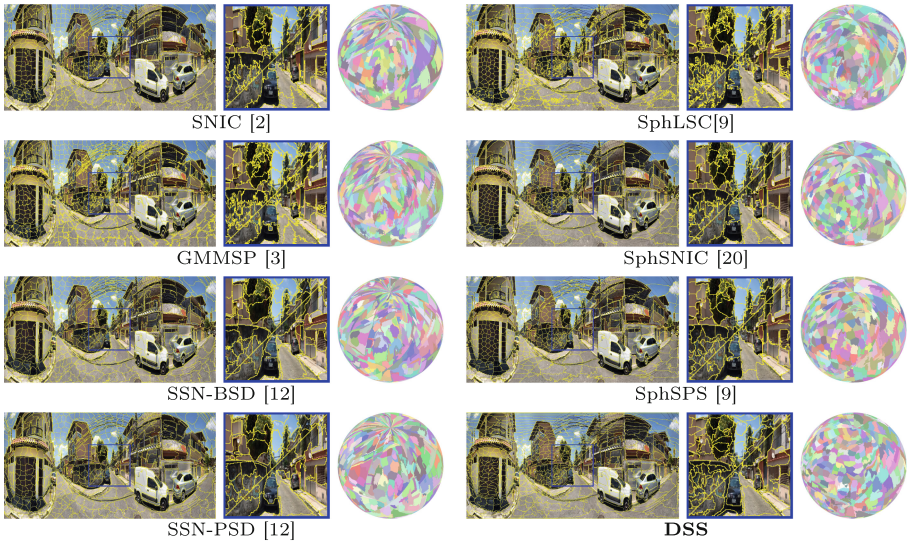


Fig. 8. Qualitative comparison on a WP image for planar (left) and spherical methods (right) for two superpixel numbers $K = 1200$ (top-left) and $K = 400$ (bottom right). Note how DSS is able to capture the car in the image center.

ularity around the sphere’s poles. DSS produces spherically regular superpixels that well capture the image objects.

4 Conclusion

In this work, we proposed DSS, the first deep learning-based spherical superpixel segmentation method. The proposed approach leverages on spherical CNN architectures dedicated to omnidirectional images having a circular geometry. We demonstrated that combining a deep learning strategy that respects the spherical geometry along with appropriate data augmentation enables to achieve higher and more robust segmentation performance than both traditional and deep learning-based methods.

We firmly believe that the presented work holds significant value for the community, given the importance of achieving both accurate segmentation and high regularity in the acquisition space, here spherical, for an effective display and processing of adjacent relationships in computer vision preprocessing tasks.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**, 2274–2282 (2012)
2. Achanta, R., Süsstrunk, S.: Superpixels and polygons using simple non-iterative clustering. In: *Conference on Computer Vision and Pattern Recognition* (2017)
3. Ban, Z., Liu, J., Cao, L.: Superpixel segmentation using gaussian mixture model. *IEEE Trans. Image Process.* **27**(8), 4105–4117 (2018)
4. Chang, A., et al.: Matterport3D: learning from RGB-D data in indoor environments. In: *International Conference on 3D Vision* (2017)
5. Chen, J., Li, Z., Huang, B.: Linear spectral clustering superpixel. *IEEE Trans. Image Process.* **26**, 3317–3330 (2017)
6. Cohen, T.S., Geiger, M., Köhler, J., Welling, M.: Spherical CNNs. In: *International Conference on Learning Representations* (2018)
7. Cordts, M., et al.: The Cityscapes dataset for semantic urban scene understanding. In: *Conference on Computer Vision and Pattern Recognition* (2016)
8. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? The KITTI vision benchmark suite. In: *Conference on Computer Vision and Pattern Recognition* (2012)
9. Giraud, R., Pinheiro, R.B., Berthoumieu, Y.: Generalization of the shortest path approach for superpixel segmentation of omnidirectional images. *Pattern Recogn.* **142**, 109673 (2023)
10. Giraud, R., Ta, V.T., Papadakis, N.: Evaluation framework of superpixel methods with a global regularity measure. *J. Electron. Imaging* **26**(6), 061603–061603 (2017)
11. Giraud, R., Ta, V.T., Papadakis, N.: Robust superpixels using color and contour features along linear path. *Comput. Vis. Image Underst.* **170**, 1–13 (2018)
12. Jampani, V., Sun, D., Liu, M.Y., Yang, M.H., Kautz, J.: Superpixel sampling networks. In: *European Conference on Computer Vision* (2018)

13. Liu, M.Y., Tuzel, O., Ramalingam, S., Chellappa, R.: Entropy rate superpixel segmentation. In: *Conference on Computer Vision and Pattern Recognition* (2011)
14. Lo, S.C.B., Li, H., Wang, Y., Kinnard, L., Freedman, M.T.: A multiple circular path convolution neural network system for detection of mammographic masses. *IEEE Trans. Med. Imaging* **21**(2), 150–158 (2002)
15. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: *International Conference on Computer Vision* (2001)
16. Schubert, S., Neubert, P., Pöschmann, J., Protzel, P.: Circular convolutional neural networks for panoramic images and laser data. In: *IEEE Intelligent Vehicles Symposium* (2019)
17. Shi, B., Bai, S., Zhou, Z., Bai, X.: DeepPano: deep panoramic representation for 3-D shape recognition. *IEEE Signal Process. Lett.* **22**(12), 2339–2343 (2015)
18. da Silveira, T.L.T., Jung, C.R.: Dense 3D scene reconstruction from multiple spherical images for 3-DoF+ VR applications. In: *IEEE Conference on Virtual Reality and 3D User Interfaces* (2019)
19. da Silveira, T.L.T., Pinto, P.G.L., Murrugarra-Llerena, J., Jung, C.R.: 3D scene geometry estimation from 360 imagery: a survey. *ACM Comput. Surv.* **55**(4), 1–39 (2022)
20. da Silveira, T.L., de Oliveira, A.Q., Walter, M., Jung, C.R.: Fast and accurate superpixel algorithms for 360° images. *Signal Process.* **189**, 108277 (2021)
21. Sun, C., Hsiao, C.W., Sun, M., Chen, H.T.: HorizonNet: learning room layout with 1D representation and pano stretch data augmentation. In: *Conference on Computer Vision and Pattern Recognition* (2019)
22. Wan, L., Xu, X., Zhao, Q., Feng, W.: Spherical superpixels: benchmark and evaluation. In: *Asian Conference on Computer Vision* (2018)
23. Wang, T.H., Huang, H.J., Lin, J.T., Hu, C.W., Zeng, K.H., Sun, M.: Omnidirectional CNN for visual place recognition and navigation. In: *International Conference on Robotics and Automation* (2018)
24. Wong, T.T., Luk, W.S., Heng, P.A.: Sampling with Hammersley and Halton points. *J. Graph. Tools* **2**(2), 9–24 (1997)
25. Xiao, J., Ehinger, K.A., Oliva, A., Torralba, A.: Recognizing scene viewpoint using panoramic place representation. In: *Conference on Computer Vision and Pattern Recognition* (2012)
26. Yang, F., Sun, Q., Jin, H., Zhou, Z.: Superpixel segmentation with fully convolutional networks. In: *Conference on Computer Vision and Pattern Recognition* (2020)
27. Yang, K., Hu, X., Fang, Y., Wang, K., Stiefelhagen, R.: Omnisupervised omnidirectional semantic segmentation. *IEEE Trans. Intell. Transp. Syst.* **23**, 1184–1199 (2020)
28. Yang, K., Zhang, J., Reiß, S., Hu, X., Stiefelhagen, R.: Capturing omni-range context for omnidirectional segmentation. In: *Conference on Computer Vision and Pattern Recognition* (2021)
29. Yogamani, S., et al.: WoodScape: a multi-task, multi-camera fisheye dataset for autonomous driving. In: *International Conference on Computer Vision* (2019)
30. Zhao, Q., Dai, F., Ma, Y., Wan, L., Zhang, J., Zhang, Y.: Spherical superpixel segmentation. *IEEE Trans. on Multimedia* **20**(6), 1406–1417 (2018)



External Prompt Features Enhanced Parameter-Efficient Fine-Tuning for Salient Object Detection

Wen Liang¹, Peipei Ran², Mengchao Bai², Xiao Liu², P. Bilha Githinji¹,
Wei Zhao², and Peiwu Qin¹(✉)

¹ Tsinghua Shenzhen International Graduate School, Tsinghua University,
Shenzhen, China

pwqin@sz.tsinghua.edu.cn

² Central Media Technology Institute, Huawei, Shenzhen, China

Abstract. Salient object detection (SOD) aims at finding the most salient objects in images and outputs pixel-level binary masks. Transformer-based methods achieve promising performance due to their global semantic understanding, crucial for identifying salient objects. However, these models tend to be large and require numerous training parameters. To better harness the potential of transformers for SOD, we propose a novel parameter-efficient fine-tuning method aimed at reducing the number of training parameters while enhancing the salient object detection capability. Our model, termed **EX**ternal Prompt features **Enhanced adapteR** Tuning (**ExPert**), features an encoder-decoder structure with adapters and injectors interspersed between the layers of a frozen transformer encoder. The adapter modules adapt the pre-trained backbone to SOD while the injector modules incorporate external prompt features to enhance the awareness of salient objects. Comprehensive experiments demonstrate the superiority of our method. Surpassing former state-of-the-art (SOTA) models across five SOD datasets, ExPert achieves 0.215 mean absolute error (MAE) in the ECSSD dataset with 80.2M trained parameters, 21% better than SelfReformer [31] and 47% better than EGNNet [33].

Keywords: Salient object detection · Segmentation · Adapter tuning · Prompt tuning · Vision language model

1 Introduction

1.1 Motivation

Salient object detection (SOD) is a widely studied task in computer vision that outputs a binary mask of the visually salient objects in an image. The detection

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78347-0_6.

of salient objects can benefit various computer vision tasks, such as semantic segmentation, instance segmentation, and object detection. In recent years, convolutional neural network (CNN) based models and transformer-based models have shown promising performances for SOD. However, although transformer-based models [15, 22, 31] generally outperform their CNN counterparts, they are more computationally expensive due to their typically large number of parameters that are essential for achieving superior performance.

The encoder-decoder framework is widely used for salient object detection, which is defined as a binary semantic segmentation. Firstly, a vision encoder is initialized with pre-trained model weights from classification or segmentation models. The next step is to fine-tune the encoder and decoder on salient object detection datasets to extend the model to the SOD task. The predicted salient masks are generated by the specific decoder with the extracted features. Beyond fine-tuning, training the pre-trained backbone along with other new sophisticated modules can gain better performance. However, it necessitates an even larger number of trained parameters.

To fine-tune pre-trained transformer models efficiently with fewer parameters, we leverage adapter tuning [4] that selectively fine-tunes certain side connections within frozen transformer blocks, facilitating transferability to downstream tasks. However, only manipulating features of the frozen backbone does not effectively tackle the salient object detection task. In [10], some learnable prompt vectors are added to the transformer layers to fine-tune large transformer models for specific tasks. Inspired by the effectiveness of visual prompt tuning, we assume that features from external backbones can be employed as prompt features. The injection of suitable external prompt features can enhance the performance of SOD models in addition to adapter tuning.

1.2 Methods Overview

We propose **EX**ternal **P**rompt features **Enhanced adapteR** **T**uning (ExPert) model to parameter-efficiently tune pre-trained transformer backbones for salient object detection. ExPert is a backbone-agnostic model and can be extended to any transformer-based pre-trained backbones. Inspired by [4, 17], ExPert uses the block-level¹ adapter module to tune the transformer backbone between each block unit. We denote the adapter of ExPert as E-adapter.

We also design a block-level injector module E-injector to receive external prompt features and inject them into the backbone so as to enhance salient features. The encoder backbone is frozen during training while the E-adapters, the E-injectors and the decoder are trained. The vision features from DINO [3], ViT [6] and BLIP [13] are chosen to verify the compatibility of our E-injector.

Moreover, we hypothesize that the captions of images are highly related to the salient elements. Based on this premise, ExPert interacts BLIP’s visual

¹ In [4], the adapter is a side connection of the feed-forward function inside the transformer block which is denoted as “FF-level”. In [17] and our ExPert, the adapter is a side connection between transformer blocks and is denoted as “block-level”.

features and text embeddings of corresponding captions using cross attention. The best result was achieved by injecting the interacted features combined with ViT’s features into the backbone. Comprehensive experiments show that ExPert surpasses CNN-based SOTA models largely and performs better than previous transformer-based models.

1.3 Contributions

Our main contributions lie in three aspects:

- We propose the ExPert model to parameter-efficiently fine-tune pre-trained transformer backbones for salient object detection. ExPert is backbone-agnostic and can use different transformer-based backbones. Comprehensive experiments demonstrate the superiority of ExPert.
- We design the block-level E-adapter to parameter-efficiently adapt the pre-trained transformer backbones to salient object detection. The size of the trained parameters of ExPert is only 80.2M.
- Our E-injector can receive different external prompt features and inject them to guide the backbone to extract salient features. Experiments demonstrate that the injection of features that contain rich semantic information largely boosts the performance.

2 Related Work

2.1 Salient Object Detection

CNN-based models are proficient at extracting local details. EGNNet [33] focuses on the complementarity between edges and the content of salient objects by extracting edge information. U2Net [19] proposes a nested U-shape convolutional network to handle inputs with flexible sizes without any pre-training. Although requiring more computing costs, transformer-based models surpass CNN-based models in SOD because transformer models can grasp the long-range semantic context of input images. SelfReformer [31] adopts a global branch to refine the local context branch with a multi-stage transformer backbone to achieve better long-range information extraction. EVP [17] fine-tunes SegFormer [26] with patch embedding prompts and Fourier transformation prompts to better differentiate objects.

2.2 Adapter Tuning and Visual Prompt Tuning

Adapter tuning is a method to fine-tune pre-trained models which was first proposed in [21] as a trainable side connection branch for parameter-efficient tuning. Later Houlsby et al. [9] used adapter tuning to parameter-efficiently train transformer-based language models. AdapterFormer [4] applies adapter tuning to Vision Transformer and achieves promising performance in multi-label classification. EVP [17] demonstrates that adapter tuning can effectively transfer

pre-trained transformer-based models to downstream tasks such as salient object detection, camouflaged object detection, and other binary segmentation tasks.

Prompt [16] is originally used in natural language processing (NLP) to instruct pre-trained language models to understand and shift to new tasks. Prompt tuning has also developed rapidly in the computer vision (CV) domain. An input-agnostic visual perturbation prompt is learned and fed to a model together with input images to repurpose pre-trained models to downstream tasks in [2]. Some learnable parameters are injected into the transformer’s input space to efficiently fine-tune large-scale transformer models in [10]. ViT-Adapter [5] uses side branches to inject spatial priors into ViT to fine-tune the model for detection and segmentation tasks. These works all show that the injection of prompt information into the original backbone can guide pre-trained models for versatile downstream tasks.

3 Methods

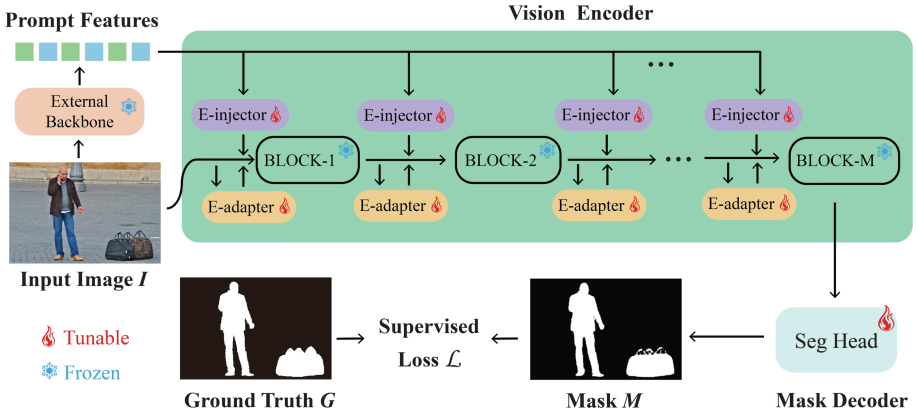


Fig. 1. The overall architecture of ExPert. During training, the vision encoder is frozen; only the E-adapters, E-injectors and the decoder are trained.

3.1 Overview

Salient object detection is an important task in the computer vision field, which detects the most salient objects in an RGB image and outputs binary masks of these objects. Let $I \in \mathbb{R}^{H \times W \times 3}$ denote the input image and $G \in \mathbb{R}^{H \times W \times 1}$ the corresponding ground truth binary mask. The output binary mask of the model is $M \in \mathbb{R}^{H \times W \times 1}$. Suppose the SOD model is \mathcal{F} and its parameters are θ , then the mask is calculated as $\mathcal{F}(I, \theta)$. The loss function \mathcal{L} in ExPert is a combination of binary cross entropy (BCE) loss and intersection over union (IoU) loss [29]. The training target is to minimize $\mathcal{L}(M, G)$ between M and G .

We propose an encoder-decoder model denominated as **EX**ternal **P**rompt features **Enhanced adapteR** **T**uning (ExPert) with block-level E-adapter and

E-injector. The architecture of ExPert is shown in Fig. 1 and entails a vision encoder, a mask decoder, some E-adapter modules between the transformer blocks of the vision encoder and some shared E-injectors for each feature scale.

Since SOD is defined as a segmentation task that is similar to semantic segmentation, pre-trained transformer backbones for segmentation or classification are preferable. A multi-scale encoder of SegFormer [26] and its decoder are chosen as the backbone and the segmentation head of ExPert. E-adapter is a lightweight side connection module that helps to transfer the pre-trained transformer backbone to salient object detection. In addition, E-injector is a lightweight side connection module that projects vision features from other backbones as guiding prompts and injects these prompts into the encoder. The detailed structures of E-adapter and E-injector are illustrated in Fig. 3.

ExPert is trained in an end-to-end manner with image-mask pairs. During training, the vision encoder is frozen while the E-adapters, E-injectors and decoder are trained. Experiments demonstrate that the combined prompts of BLIP and ViT achieve the best result. As a backbone-agnostic model, ExPert can switch between different transformer backbones with simple modification² while the decoder needs to be specified according to different backbones.

3.2 Encoder and Decoder

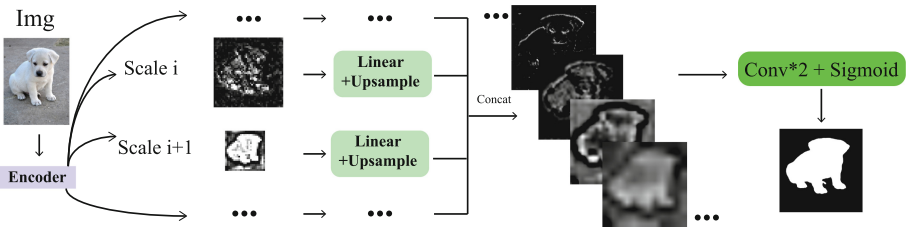


Fig. 2. The decoder of ExPert for multi-scale features. The illustration of feature images is visualized by choosing a random slice of the channel dimension. ExPert’s final mask is generated by resizing this mask to the original size.

Transformer backbones can be classified into two types according to whether the scale of features changes. One is the single-scale backbone and the other is the multi-scale backbone. The features of the single-scale backbone keep the same size during the forward propagation while the multi-scale backbone’s features change size. For salient object detection, former research [19, 31, 34] emphasized the importance of multi-scale feature fusion to get finer segmentation masks. Since multi-scale features are crucial to dense prediction for finer details, multi-scale backbones have an advantage over single-scale backbones in segmentation tasks. Therefore we choose the MiT-B4 version of SegFormer [26] as our multi-scale backbone. The encoder of SegFormer has 4 stages of feature extraction

² More details can be found in the supplementary materials.

and each stage has a different scale of features. The decoder is kept intact as the original one as shown in Fig. 2.

For each stage of feature extraction, denote the output feature of the i_{th} stage as F_i^{out} . A linear layer L_i projects F_i^{out} to align the channel dimension for the following concatenation, followed by a bilinear upsampling $\mathcal{U}()$ to align the spatial scale of features. Then all aligned features from different stages are concatenated by $Cat()$ to get F_c . Finally, two convolution layers, C_{fuse} , C_{pred} and the sigmoid function $\mathcal{S}()$ are applied for the mask generation. The mask M is generated as Eq. 2.

$$F_c = Cat\left(\sum_{i=1}^I \mathcal{U}(L_i(F_i^{out}))\right) \quad (1)$$

$$M = \mathcal{S}(C_{pred}(C_{fuse}(F_c))) \quad (2)$$

3.3 E-adapter

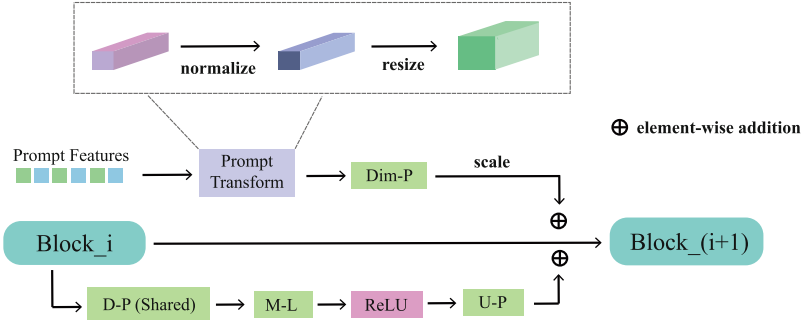


Fig. 3. The detailed structure of E-adapter and E-injector. D-P is the down projection layer, M-L is the median linear layer, U-P is the up projection layer and Dim-P is the dimension projection layer.

To fine-tune the transformer backbone in a parameter-efficient manner, we employ the adapter tuning method, which reduces the dimension of features through a bottleneck design, thereby diminishing the number of trained parameters. As shown in the bottom branch of Fig. 3, we propose the E-adapter, consisting of a down-projection(D-P) layer with parameters P_e^d , a median linear(M-L) layer L_e^m and an up-projection(U-P) layer with parameters P_e^u . It is noteworthy that the D-P layers of the E-adapter are shared for features of the same spatial scale in order to further diminish the number of trained parameters. The M-L and U-P layers are independent for each E-adapter. The additional low-dimension M-L layers serve to increase the variability of the E-adapter. Define

the i_{th} block forward function as $\mathcal{B}_i()$, the $(i + 1)_{th}$ block’s feature F_{i+1} after the E-adapter is computed as Eq. 4.³

$$F_i^{ad} = P_e^u \cdot ReLU(L_e^m(P_e^d \cdot F_i)) \quad (3)$$

$$F_{i+1} = \mathcal{B}_{i+i}(F_i + F_i^{ad}) \quad (4)$$

3.4 E-injector

Previous works [5, 17] show that side connection modules like adapters can introduce extra information to boost the model’s performance in object detection and segmentation tasks. We design the E-injector to inject external features from other backbones of the same input images as guiding prompts into the encoder for salient object detection. The E-injector’s structure is depicted in the top branch of Fig. 3.

If the number of injected prompt features is J , the j_{th} E-injector is composed of a prompt transformation $Trans_j()$ and a dimension projection (Dim-P) layer with parameters P_j^{dim} . Since visual prompts might vary in size and shape, a feature transformation $Trans_j()$ fits the prompt feature F'_j to the i_{th} layer’s feature F_i of the frozen backbone. $Trans_j()$ is composed of a normalization and a resize operation. The E-injector can receive different transformer features F'_j including the features from DINO, ViT and BLIP’s vision encoder. The output of the E-injector F_j^{inj} is generated as Eq. 5. To better adjust the injector features to the backbone, a learnable scaling vector α_j is used to weight them. The $(i + 1)_{th}$ block’s feature F_{i+1} is computed as Eq. 6.

$$F_j^{inj} = P_j^{dim} \cdot Trans_j(F'_j) \quad (5)$$

$$F_{i+1} = \mathcal{B}_{i+i}(F_i + F_i^{ad} + \sum_{j=1}^J F_j^{inj} \times \alpha_j) \quad (6)$$

Finding suitable prompt features is crucial to the quality of salient masks for E-injector. DINO [3] is a self-supervised model trained without labels that exhibits an obvious tendency to focus on objects in an image. Observing the attention maps of DINO reveal coarse masks that are nearly similar to the masks of the salient objects, we assume this kind of object-aware features can aid pre-trained models in locating objects. Besides, considering that our multi-scale backbone has fewer layers of high resolution features compared to single-scale backbones such as ViT, our backbone’s overall perception of an image might be complemented by the features of ViT’s last layer. ExPert uses the features of ViT/B-16 as the auxiliary features for better global perception of images.

While piling up the features might strengthen the visual details, it is still hard to guide the model to recognize the notion of saliency. It is noteworthy that the caption naturally contains the descriptions of the salient objects in an image which are beneficial to SOD. Therefore, we envision that the caption of an image is highly related to the salient objects and contains the salient

³ The $P \cdot F$ in Eq. 3 represents the linear projection to features F with parameters P.

information. Some Vision Language Models (VLM) like BLIP [13] are trained on large caption datasets.⁴ The features from BLIP’s vision encoder trained with image-text labels, which contain rich semantic information of an image, are injected as semantic-enhanced features by E-injector.

To fully explore the rich semantic information in BLIP, we interact BLIP’s vision features and BLIP’s captions for better focus on salient objects. Although there are no captions or other text information in SOD datasets, we can generate them by the inference results of the BLIP model. For an image I , the corresponding caption is generated by BLIP using beam search. The caption is tokenized and embedded by BLIP’s text encoder to get the text embedding T . The interacted feature is acquired via the cross-attention between the last layer vision feature V_b of BLIP and T . In each cross attention layer, V_b is projected as the query and T is projected as the key and value by linear projections. Experiments show that one cross-attention layer is enough for the interaction⁵. Since BLIP is trained with image-caption pairs, the alignment of image and text features is not of pixel scale but rather of patch scale. As auxiliary prompt features, too many cross attentions might lead to unexpected noises that harm the final performance.

Since the features from ViT and similar models can refine the mask in detail and the features from BLIP can inject semantic information into ExPert, we combine these two features together for E-injector. The final version of ExPert uses the features from ViT’s last layer and the interacted features by cross-attention from BLIP. Experiments show that the combination prompt version achieves the best performance. Unless specified, ExPert represents our best version.

4 Experiments

4.1 Experiment Setup

Implemented with PyTorch, ExPert is trained on the DUTS [24] dataset with a batch size of eight using two V100 16G GPUs. The encoder and the decoder are initialized with the publicly released pre-trained weights of SegFormer and the other parameters are initialized randomly. We used the AdamW optimizer and the learning rate is set to $2e-4$ with a weight decay of zero.

We used ECSSD [27], DUT-OMRON [28], HKU-IS [11] and PASCAL-S [14] as the evaluation datasets. Four metrics are adopted for our model evaluation: the mean absolute error (MAE), the F-measure F_β [1], the maximum E-measure [8] and the S-measure [7]. More details on implementation, datasets and metrics can be found in the supplement file.

⁴ CLIP [20] is a well-known VLM model trained with millions of image-text pairs. However the text of CLIP is a simple sentence with the class name which is not the caption of the whole image. The resolution of CLIP’s training images is $224*224$, the feature is $7*7$ with the patch size of 32 which is too small to upsample. Therefore ExPert does not consider CLIP’s features.

⁵ More details can be found in the supplementary file.

4.2 Comparison with SOTA Models

Table 1. The quantitative metrics of our best version of ExPert (ViT & BLIP+ Injection) and four SOTA models. Best results are in **bold**. SR is the abbreviation of SelfReformer. EVP and SR are two transformer-based SOTA models while EGNet and U2Net are two CNN-based SOTA models. The column of trained parameters (TP) shows the size of trained parameters of each model.

Methods	DUTS-TE				DUT-OMRON				ECSSD			
	MAE↓	FM↑	EM↑	SM↑	MAE↓	FM↑	EM↑	SM↑	MAE↓	FM↑	EM↑	SM↑
EGNet	.0431	.8507	.9148	.8775	.0564	.7686	.8640	.8345	.0405	.9293	.9494	.9192
U2Net	.0443	.8477	.9102	.8737	.0544	.7930	.8794	.8466	.0330	.9408	.9572	.9276
EVP	.0297	.9033	.9521	.9016	.0485	.8195	.9047	.8529	.0303	.9475	.9636	.9335
SR	.0266	.9016	.9514	.9110	.0433	.8058	.8899	.8603	.0273	.9480	.9651	.9356
ExPert	.0231	.9158	.9594	.9179	.0429	.8399	.9101	.8711	.0215	.9550	.9707	.9422

(a)

Methods	HKU-IS				PASCAL-S				TP
	MAE↓	FM↑	EM↑	SM↑	MAE↓	FM↑	EM↑	SM↑	Size
EGNet	.0345	.9160	.9520	.9098	.0821	.8166	.8673	.8469	412.3M
U2Net	.0312	.9238	.9539	.9160	.0817	.8097	.8609	.8414	168.1M
EVP	.0253	.9426	.9694	.9294	.0674	.8486	.8930	.8701	14.1M
SR	.0241	.9406	.9689	.9309	.0600	.8513	.8978	.8807	349.7M
ExPert	.0198	.9498	.9747	.9375	.0538	.8670	.9099	.8932	80.2M

(b)

Quantitative Comparison. We compare our model’s salient masks with four representative state-of-the-art models on five salient object detection datasets: DUTS-TR, DUT-OMRON, ECSSD, HKU-IS and PASCAL-S. Two CNN SOTA models EGNet [33] and U2Net [19] are considered together with two transformer SOTA models SelfReformer [31] and EVP [17]. The metrics are calculated under the same condition using the prediction masks of different models⁶. The prediction masks are all provided by the official release⁷.

The results are shown in Table 1. Our method achieves the best performance across all the SOTA models on all five datasets, which demonstrates the effectiveness of ExPert and the potential of the transformer backbone on salient

⁶ We use the public codes of [SOD_Evaluation_Metrics](#) to compute the metrics.

⁷ Considering that EVP’s official mask is 352*352 which is not the original size, we resize the prediction map of EVP to the size of ground truth and then compute the metrics.

object detection. For the MAE metric, ExPert surpasses the second best Self-Reformer by 0.0058 (around 21% improvement) in the ECSSD dataset and surpasses EGNet by around 47%. The superiority on all other three metrics demonstrates that ExPert has stronger competence to segment the salient objects in an image. Regarding the size of trained parameters, ExPert is more parameter efficient than all SOTA models except EVP whose trained parameters are also under 100 M as ExPert but with a smaller size. Therefore, ExPert realizes a good trade-off between performance and the size of trained parameters. Figure 4 shows the F-measure curves and the precision-recall curves of ExPert and 4 SOTA models on five datasets. It is observable in these curves that our model consistently outperforms all other models.

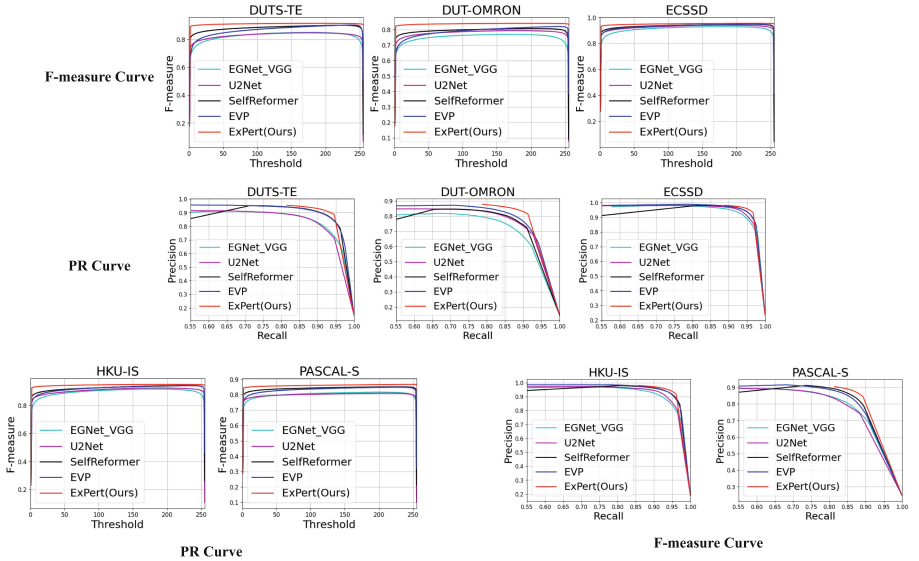


Fig. 4. The F-measure curves and the precision-recall (PR) curves of ExPert and four SOTA models on five datasets.

To ensure sufficient comparisons, ExPert is also compared to other latest SOD models⁸ including M3Net [30], DSRNet [23], TCRNet [32], BBRF model [18], IMSFNet [25] and CTD-L [12]. As shown in Table 2, ExPert performs better in three metrics than these latest SOD models in the DUT-OMRON dataset, which demonstrates the superiority of ExPert.

⁸ Due to the absence of codes or salient maps of some models, we directly use the metrics results in the published paper. The results of M3Net [30] are calculated using the official salient maps of the M3Net SwinB version.

Table 2. The results on the DUT-OMRON dataset of different SOD models and ExPert of metrics MAE, F_β , max E-measure and S-measure. The best results are in bold.

Metrics	ExPert	M3Net [30]	DSRNet [23]	TCRNet [32]	BBRF [18]	IMSFNet [25]	CTD-L [12]
MAE↓	.042	.045	.051	.054	.042	.053	.049
max-FM↑	.839	.832	.810	0.791	.814	.760	.789
max-EM↑	.910	.902	/	/	.887	.777	.881
SM↑	.871	.872	.852	0.843	.855	/	/



Fig. 5. The qualitative results of ExPert and four SOTA models. From left to right are the images, the ground truths, ExPert’s masks, EVP’s masks, SelfReformer’s masks, U2Net’s masks and EGNet’s masks. Better visual effect when zooming in.

Qualitative Comparison. In Fig. 5, we show the qualitative comparison between our model and SOTA models to give readers an intuitive comprehension. Compared to SOTA models, ExPert’s masks are more accurate in details and can distinguish some ambiguous scenarios like reflection in water in the second row. The head of the girl is similar to the tree in the background in the first row while the hair and the cloth similar to the dark background are confusing in the fifth row. Four SOTA models can’t differentiate these nuances but ExPert

can handle the ambiguity. In the first row, ExPert’s mask is even more accurate than the ground truth which contains all the hair. Moreover, the semantic information injection from BLIP aids ExPert in recognizing the relationship between objects in an image. For example, in the fourth row, the baggage on the ground is obviously related to the man. ExPert segments them out while some other SOTA models neglect the baggage or focus wrongly on the street lamp. Another example is the third row, the children in the car should be regarded as a whole with the car. Semantic information also assists ExPert in handling complex scenarios, such as shadow interference or color similarity, for instance, the case of discerning the body of the deer in the shadow. As for salient objects of small sizes, ExPert can well recognize them with clear details and less ambiguity as shown in the last two rows.

4.3 Ablation Study

Tuning Methods: To verify the effectiveness of E-adapter on fine-tuning models for salient object detection, the performances of different fine-tuning methods are evaluated on the ECSSD dataset. As shown in Table 3, full fine-tuning⁹ achieves the best performance but requires a large size of trained parameters. Training from scratch requires a large size of trained parameters with poor performances and head tuning¹⁰ doesn’t output satisfying results either. As for adapter tuning, the performances are close to the fully tuned version with much fewer trained parameters.

Single-scale and Multi-scale Backbone: To compare the performance of the single-scale backbone and the multi-scale backbone, all fine-tuning methods are used to fine-tune MiT-B4 SegFormer and two of them are used to fine-tune ViT-B/16. The decoder of ViT is the same as [35], more details can be found in our supplement file. In Table 3, training from scratch and adapter tuning of ViT are worse than the counterparts of SegFormer because multi-scale features can extract more detailed information in the images. Therefore, ExPert uses multi-scale pre-trained models as the encoder backbone for their finer details.

Adapter Level: In [4], the adapter module is side connected inside the transformer block while in ExPert we use the block-level adapter, which is side connected between transformer blocks. In Table 3, the block-level adapter performs better than the FF-level adapter which demonstrates that in the segmentation task the block-level adapters are more advantageous. We suppose that the self-attention layers are also important in image recognition and should also be covered by adapters.

⁹ The full fine-tuning method trains all the parameters of the backbone and decoder using the new datasets.

¹⁰ The head tuning method trains only the decoder while keeping the backbone frozen.

Table 3. The results on ECSSD datasets of different fine-tune methods. Scratch refers to training from scratch. Head tune and full tune denote head tuning and full fine-tuning respectively. Adapter tune uses the block-level E-adapter to adapter tune pre-trained models. “-S” means the SegFormer backbone. “-V” means the ViT-B/16 backbone. “-FF” means the adapter is of FF level, otherwise the block-level. The metric in **bold** is the best.

Methods	MAE↓	max-FM↑	max-EM↑	SM↑	Trained Parameters
Scratch-S	0.0784	0.8552	0.8758	0.8085	733.8MB
Head tune-S	0.0740	0.8922	0.9292	0.8444	36.1MB
Full tune-S	0.0282	0.9501	0.9679	0.9320	733.8MB
Adapter tune-S	0.0354	0.9447	0.9626	0.9193	50.9MB
Adapter tune-S-FF	0.0383	0.9446	0.9612	0.9152	48.0MB
From scratch-V	0.0312	0.9412	0.9648	0.9264	1093.7MB
Adapter tune-V	0.0362	0.9300	0.9569	0.9163	130.3MB

Prompt Feature: To verify the effectiveness of E-injector and different prompt features, we denote the baseline model as the multi-scale encoder with E-adapter but without E-injector. In Table 4, we compared E-injector with five prompt features to the baseline. Injecting DINO features as prompt features performs better than the baseline, which shows that object-aware features from DINO can guide the encoder to focus more on salient objects. The ViT features injection is better than the DINO features injection, suggesting that pre-trained models’ semantic information can further boost the performance of SOD. Additionally, although the multi-scale backbone can extract multi-scale salient features, its global layers are usually shallower than those in single-scale backbones. This weakness can be alleviated by the injection of ViT’s features which go through more layers of the full-size scale.

Table 4. The results on ECSSD datasets of different prompt features for E-adapter. BLIP+ represents the interacted features after cross-attention of BLIP. The baseline only uses the E-adapter in 3.3 without the E-injector.

Methods	MAE↓	max-FM↑	max-EM↑	SM↑	Trained Parameters
Baseline	0.0354	0.9447	0.9626	0.9193	50.9MB
DINO Inject	0.0298	0.9473	0.9625	0.9330	60.0MB
ViT Inject	0.0272	0.9497	0.9654	0.9357	60.0MB
BLIP Inject	0.0249	0.9516	0.9681	0.9380	60.0MB
BLIP+ Inject	0.0231	0.9534	0.9689	0.9408	80.2MB
ViT & BLIP+ Inject	0.0215	0.9550	0.9707	0.9422	80.2MB

Compared to the injection of ViT’s features, the injection of BLIP’s vision feature performs better. Since BLIP is trained with image-text pairs, the seman-

tic recognition of BLIP is stronger than ViT which is trained with image-label pairs. This indicates that semantic information is critical for the model to detect salient objects. The BLIP+ injection version surpasses the BLIP injection version, verifying that our cross-attention interaction between BLIP’s image features and caption embeddings successfully highlight the salient regions. The best performance comes from the combination of ViT’s features and the BLIP+ features, which captures both the detailed large scale features and the rich semantic information.

5 Conclusion

We introduce the **EX**ternal **P**rompt features **Enhanced adapteR** **T**uning (ExPert) model, designed to efficiently fine-tune pre-trained transformer models for salient object detection. E-adapter efficiently tailors pre-trained backbones to extract salient features, while E-injector integrates various external features as guiding prompts, enhancing the localization of salient objects. Additionally, to enhance the representation of fine details, ExPert incorporates ViT features into the backbone to complement shallow global layers. Furthermore, to capture the relationship between image content and salient elements, the image-text interaction features from BLIP are integrated into the encoder, enabling better differentiation of complex scenarios. Comprehensive experiments demonstrate the superior performance of our ExPert over both state-of-the-art CNN-based and transformer-based models across five validation datasets.

Looking ahead, further enhancement of ExPert may include exploration of additional prompt features. It is possible to inject other prompt features such as color or texture information into the backbone. Moreover, ExPert’s paradigm could also be applied to other segmentation tasks, such as semantic segmentation and panoptic segmentation. We also find that ExPert’s performance is influenced by the quality of the generated captions by BLIP. To make ExPert more robust, how to filter out captions of low quality is a challenge. We leave these possible directions for future research.

References

1. Achanta, R., Hemami, S., Estrada, F., Susstrunk, S.: Frequency-tuned salient region detection. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1597–1604. IEEE (2009)
2. Bahng, H., Jahanian, A., Sankaranarayanan, S., Isola, P.: Exploring visual prompts for adapting large-scale models. arXiv preprint [arXiv:2203.17274](https://arxiv.org/abs/2203.17274) (2022)
3. Caron, M., et al.: Emerging properties in self-supervised vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 9650–9660 (2021)
4. Chen, S., et al.: Adaptformer: adapting vision transformers for scalable visual recognition. In: Advances in Neural Information Processing Systems, vol. 35, pp. 16664–16678 (2022)

5. Chen, Z., et al.: Vision transformer adapter for dense predictions. arXiv preprint [arXiv:2205.08534](https://arxiv.org/abs/2205.08534) (2022)
6. Dosovitskiy, A., et al.: An image is worth 16×16 words: transformers for image recognition at scale. arXiv preprint [arXiv:2010.11929](https://arxiv.org/abs/2010.11929) (2020)
7. Fan, D.P., Cheng, M.M., Liu, Y., Li, T., Borji, A.: Structure-measure: a new way to evaluate foreground maps. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4548–4557 (2017)
8. Fan, D.P., Gong, C., Cao, Y., Ren, B., Cheng, M.M., Borji, A.: Enhanced-alignment measure for binary foreground map evaluation. arXiv preprint [arXiv:1805.10421](https://arxiv.org/abs/1805.10421) (2018)
9. Houshy, N., et al.: Parameter-efficient transfer learning for NLP. In: International Conference on Machine Learning, pp. 2790–2799. PMLR (2019)
10. Jia, M., et al.: Visual prompt tuning. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) European Conference on Computer Vision, pp. 709–727. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19827-4_41
11. Li, G., Yu, Y.: Visual saliency based on multiscale deep features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5455–5463 (2015)
12. Li, J., Qiao, S., Zhao, Z., Xie, C., Chen, X., Xia, C.: Rethinking lightweight salient object detection via network depth-width tradeoff. *IEEE Trans. Image Process.* **32**, 5664–5677 (2023)
13. Li, J., Li, D., Xiong, C., Hoi, S.: Blip: bootstrapping language-image pre-training for unified vision-language understanding and generation. In: International Conference on Machine Learning, pp. 12888–12900. PMLR (2022)
14. Li, Y., Hou, X., Koch, C., Rehg, J.M., Yuille, A.L.: The secrets of salient object segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 280–287 (2014)
15. Liu, N., Zhang, N., Wan, K., Shao, L., Han, J.: Visual saliency transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4722–4732 (2021)
16. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: a systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.* **55**(9), 1–35 (2023)
17. Liu, W., Shen, X., Pun, C.M., Cun, X.: Explicit visual prompting for universal foreground segmentations. arXiv preprint [arXiv:2305.18476](https://arxiv.org/abs/2305.18476) (2023)
18. Ma, M., Xia, C., Xie, C., Chen, X., Li, J.: Boosting broader receptive fields for salient object detection. *IEEE Trans. Image Process.* **32**, 1026–1038 (2023)
19. Qin, X., Zhang, Z., Huang, C., Dehghan, M., Zaiane, O.R., Jagersand, M.: U2-Net: going deeper with nested U-structure for salient object detection. *Pattern Recogn.* **106**, 107404 (2020)
20. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
21. Rebuffi, S.A., Bilen, H., Vedaldi, A.: Learning multiple visual domains with residual adapters. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
22. Ren, S., Wen, Q., Zhao, N., Han, G., He, S.: Unifying global-local representations in salient object detection with transformer. arXiv preprint [arXiv:2108.02759](https://arxiv.org/abs/2108.02759) (2021)
23. Song, X., Guo, F., Zhang, L., Lu, X., Hei, X.: Salient object detection with dual-branch stepwise feature fusion and edge refinement. *IEEE Trans. Circuits Syst. Video Technol.* **34**, 2832–2844 (2023)

24. Wang, L., et al.: Learning to detect salient objects with image-level supervision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 136–145 (2017)
25. Xia, C., Sun, Y., Fang, X., Ge, B., Gao, X., Li, K.C.: IMSFNet: integrated multi-source feature network for salient object detection. *Appl. Intell.* **53**(19), 22228–22248 (2023)
26. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: SegFormer: simple and efficient design for semantic segmentation with transformers. In: Advances in Neural Information Processing Systems, vol. 34, pp. 12077–12090 (2021)
27. Yan, Q., Xu, L., Shi, J., Jia, J.: Hierarchical saliency detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1155–1162 (2013)
28. Yang, C., Zhang, L., Lu, H., Ruan, X., Yang, M.H.: Saliency detection via graph-based manifold ranking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3166–3173 (2013)
29. Yu, J., Jiang, Y., Wang, Z., Cao, Z., Huang, T.: Unitbox: an advanced object detection network. In: Proceedings of the 24th ACM International Conference on Multimedia, pp. 516–520 (2016)
30. Yuan, Y., Gao, P., Tan, X.: M3Net: multilevel, mixed and multistage attention network for salient object detection. arXiv preprint [arXiv:2309.08365](https://arxiv.org/abs/2309.08365) (2023)
31. Yun, Y.K., Lin, W.: SelfReformer: self-refined network with transformer for salient object detection. arXiv preprint [arXiv:2205.11283](https://arxiv.org/abs/2205.11283) (2022)
32. Zhang, Q., Zhao, R., Zhang, L.: TCRNet: a trifurcated cascaded refinement network for salient object detection. *IEEE Trans. Circuits Syst. Video Technol.* **33**(1), 298–311 (2022)
33. Zhao, J.X., Liu, J.J., Fan, D.P., Cao, Y., Yang, J., Cheng, M.M.: EGNNet: edge guidance network for salient object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 8779–8788 (2019)
34. Zhao, T., Wu, X.: Pyramid feature attention network for saliency detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3085–3094 (2019)
35. Zheng, S., et al.: Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6881–6890 (2021)



Recall-Based Knowledge Distillation for Data Distribution Based Catastrophic Forgetting in Semantic Segmentation

Samiha Mirza^{1(✉)}, Apurva Gala², Pandu Devarakota², Vuong D. Nguyen¹,
Pranav Mantini¹, and Shishir K. Shah¹

¹ Quantitative Imaging Lab, University of Houston, Houston, TX, USA
smirza6@cougarnet.uh.edu

² Shell Global Solutions, Houston, TX, USA

Abstract. Semantic segmentation involves labeling each pixel in an image with a corresponding class label, enabling detailed scene understanding. In dynamic environments, where conditions change over time, incremental learning techniques are essential for updating segmentation models with newly acquired data. However, incremental segmentation faces the challenge of catastrophic forgetting, where models lose previously learned knowledge when trained on new data distributions. To address this, we propose a recall-based knowledge distillation approach for stable segmentation model training across dynamic environments. Our method combines the strengths of knowledge distillation and recall learning to enhance the model's ability to recall information from previous data distributions while adapting to new ones. By reintroducing a small portion of the previous dataset during training and applying tailored distillation techniques, our approach mitigates catastrophic forgetting and improves the robustness of these models. Through comprehensive evaluations, we demonstrate the effectiveness of our approach in two scenarios: salt segmentation in seismic datasets and tumor segmentation in MRI datasets. Our method offers a promising solution for addressing the challenges of catastrophic forgetting in incremental semantic segmentation, facilitating the development of more adaptive and reliable computer vision systems in dynamic environments.

Keywords: Segmentation · Catastrophic Forgetting · Knowledge Distillation · Recall Learning · Seismic imaging · MRI

1 Introduction

Semantic segmentation is a fundamental task in computer vision. It involves labeling each pixel in an image with a class label, thereby enabling a detailed understanding of visual scenes. Semantic segmentation has seen remarkable progress in various domains [13, 18] such as energy, medical image analysis [1, 16], autonomous driving, etc. To adapt to the dynamic environments in production,

these models are frequently updated with new data [9, 26]. This process of updating a model with new data is called incremental learning [32]. As shown in Fig. 1, a simple approach to address this is to retrain the model from scratch using all the available old and new data [19]. However, this approach requires an expensive training phase and significant storage resources for the old and new data. A more efficient way to update models is to fine-tune them incrementally on the new data, starting with the weights of the previous model. Incremental segmentation models however poses a critical challenge of ensuring that the model adapts to incoming data without compromising the knowledge acquired from previous data - a phenomenon called catastrophic forgetting [11, 22]. Catastrophic forgetting can be either class-based or distribution-based. In class-based, the model’s ability to segment classes that it had previously learned degrades when presented with new tasks. In distribution-based, the model’s ability to segment the same classes when presented with different data distributions degrades. Recent methods have addressed class-based forgetting using recall/rehearsal [20] or knowledge distillation [5]. Knowledge distillation [5] approaches have offered a promising solution to address this issue by distilling the knowledge learned from the accumulated data into a compact form onto the new task model. This enables the model to maintain performance across varying data distributions and adapt more efficiently to new tasks. Recall learning [20] involves periodically reintroducing past data samples during training to counteract the detrimental effects of catastrophic forgetting. However, research focusing on distribution-based catastrophic forgetting is limited [20]. Our proposed method offers a promising solution for addressing the challenges of catastrophic forgetting due to distribution shifts in incremental semantic segmentation. In this paper, we explore the application of knowledge distillation and recall learning for segmentation model by training them across varying data distributions. In doing so, we propose a recall-based distillation approach to address catastrophic forgetting. Our method builds upon the strengths of knowledge distillation and recall learning techniques to enhance the model’s ability to recall information from previous data distributions while adapting to new ones. By reintroducing a small portion of the previous dataset during training and applying tailored distillation techniques, our method aims to alleviate distribution based catastrophic forgetting and improve the robustness and adaptability of semantic segmentation models in real-world settings. Additionally, to the best of our knowledge, our method is the only approach that effectively combines recall learning with knowledge distillation to address distribution-based catastrophic forgetting.

Our paper makes the following contributions:

1. We propose to use distillation loss in recall learning from the old datasets to enable the model to learn to recall and reweigh the model parameters.
2. We evaluate the approach on two problem domains: salt segmentation in seismic datasets and tumor segmentation in MRI datasets
3. We investigate the impact of applying different training curriculum on the model’s performance, providing insights into the learning dynamics of incremental segmentation models.

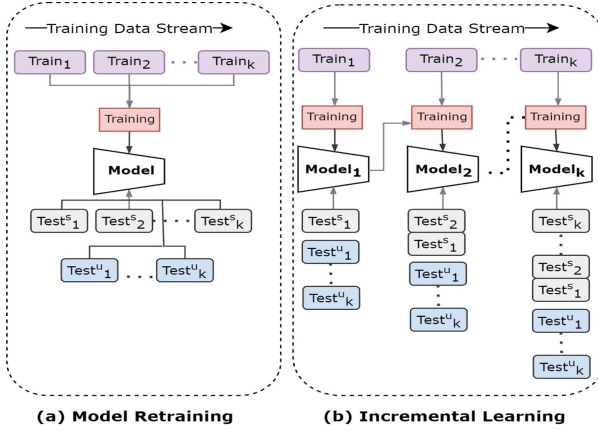


Fig. 1. Comparison of model retraining and Incremental Learning (IL). IL adapts to new data distributions over time using previous model weights, reducing training time and memory requirements, and making it more practical for updating models in dynamic environments.

2 Related Works

2.1 Incremental Learning in Semantic Segmentation

Incremental learning in Segmentation is a relatively new field that has been gaining attention in recent years [8,24]. This research mainly focuses on finding ways to prevent catastrophic forgetting [28]. It also faces the challenge of background shift, which was first pointed out in [8]. Background shift in incremental learning occurs when old classes from previous iterations are merged or collapsed into the background. Strategies such as pseudo-labelling [30] can mitigate this issue by dynamically updating the background class based on the model’s predictions and preserving the distinctions between old and new classes. Recall learning and knowledge distillation are two main techniques studied in literature.

2.2 Knowledge Distillation

Knowledge distillation (KD) approach [5], is widely being investigated to mitigate catastrophic forgetting. One of the first studies by Michieli *et al.* [25], focuses on applying distillation scheme to retain knowledge about segmenting the old classes in an incremental setting. They developed methodologies involving learning of output prediction maps, intermediate layers, and feature maps of the teacher models. Phan *et al.* [27] proposed a class similarity knowledge distillation (CSW-KD) that distills the knowledge of a previous model on old classes that are similar to the new ones. Further, Douillard *et al.* [10] proposed Local pooled output distillation (POD) that employs a multi-scale pooling distillation scheme to

maintain spatial relationships in features and n entropy-based pseudo-labelling to handle background shifts. Further studies [12, 14, 30] have also proposed KD to tackle class-based catastrophic forgetting.

2.3 Recall Learning

Recall learning involves storing and replaying past data samples or experiences during the training of a model. This technique allows the model to retain knowledge of previously seen data. Huang *et al.* [15] proposed a half-real half-fake distillation approach that includes synthetic images alongside real ones during the training of new tasks to refresh models’ memory of old classes. Yan *et al.* [31] proposed a method called RECALL which utilizes generative adversarial networks and web-crawled data to produce new samples representing old classes, which are then integrated into the training data for new tasks. Another strategy by Michieli *et al.* [25] introduces an expectation-maximization framework, merging relabeling and replay-based methods to enhance continual learning.

These studies primarily concentrate on addressing catastrophic forgetting in class-based incremental segmentation scenarios. In our work, we leverage the advantages of both knowledge distillation and recall learning to introduce a recall-based distillation approach for distribution based forgetting. Unlike existing methods, our approach specifically focuses on enhancing the model’s ability to recall information from previous datasets for individual classes. We achieve this by reintroducing a portion of the previous dataset to the incremental model and applying distillation techniques tailored to facilitate better recall in the segmentation model. Additionally, we incorporate a curriculum strategy into the training batches to systematically investigate the learning progression of the model.

3 Method

3.1 Problem Definition

The task of semantic segmentation is to assign a class c , to each pixel in a given image where the number of classes, N , typically ranges from 0 to $N - 1$, where 0 represents the background class. As depicted in Fig. 2, in an incremental learning setting, a model $Model_i : \{F(\mathbf{x}_{ij}) \rightarrow \mathbf{y}_{ij}\}_{i,j=1}^{h,w}$, where h and w denote the height and width of the image respectively, is trained incrementally. Here, \mathbf{x}_{ij} denotes the pixel at the i^{th} row and j^{th} column of the image, and \mathbf{y}_{ij} represents the ground truth segmentation map pixel. The incremental training process begins with the weights of the previous model $Model_{i-1}$ and continues training as new dataset $Train_i$ is acquired. At each step, after $Model_i$ is trained, it is tested on a set of seen datasets $\{Test_x^s\}_{x=1}^i$, where $Test_i^s$ is a non-intersecting subset of images coming from dataset $Train_i$ and also on a set of unseen datasets $\{Test_x^u\}_{x=1}^k$, k =number of unseen datasets. Catastrophic forgetting is said to occur when $Model_i$ performs poorly on the previous test dataset $Test^s$ or on

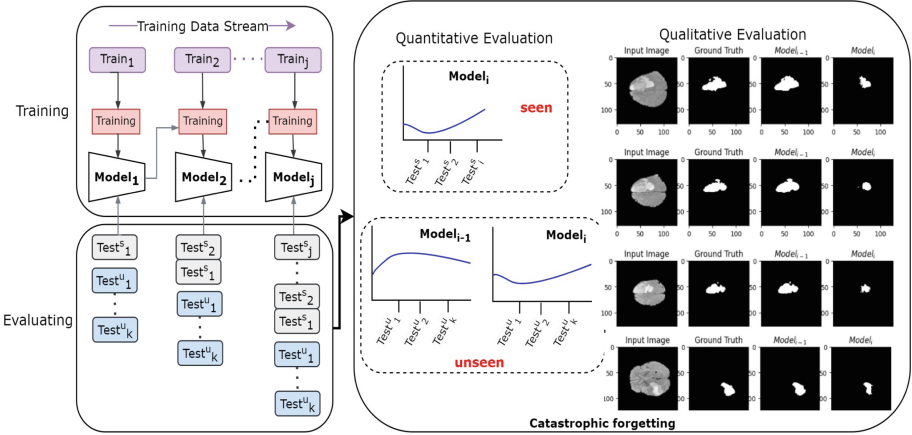


Fig. 2. Catastrophic forgetting in incremental segmentation. It can be seen quantitatively and qualitatively that when the model is updated using new data ($Train_i$), $Model_i$ performs poorly on previously seen $Test_i^s$ as well as on some unseen $Test_i^u$, indicating a loss of performance on previously learned data distributions.

a subset of $Test^u$ when updated with new training dataset $Train_i$. This study focuses on scenarios where the input distribution of the data varies across the incremental tasks while the set of classes remains fixed with $N = 2$.

3.2 Proposed Recall-Based Distillation Framework

Figure 3 illustrates our proposed recall-based distillation approach. We first train the $Model_{i-1}$ to recognize pixels belonging to classes C from training dataset $Train_{i-1}$ using the cross-entropy loss, L_{CE} .

$$L_{CE} = -\frac{1}{N} \sum_{i=1}^N (y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)), \quad (1)$$

where y_i is the true label (either 0 or 1) and \hat{y}_i is the predicted probability that the input belongs to class i . Then $Model_{i-1}$ is frozen and acts as a teacher model for the next incremental model $Model_i$. We proceed to train $Model_i$ using the new dataset $Train_i$ and $Train_{i-1}'$ which is a subset of previous dataset $Train_{i-1}$.

Batching: We batch images from $Train_{i-1}'$ and $Train_i$ separately. This step is crucial as the organization of data during training significantly impacts the model’s learning process. There are four strategies for concatenating batches from the two datasets, as illustrated in Fig. 4. One approach is to concatenate the batches randomly, providing a diverse mix of data in each batch. Alternatively, a more rigid strategy involves passing all batches from $Train_{i-1}'$ first followed by $Train_i$, or vice versa. While this method ensures exposure to both datasets

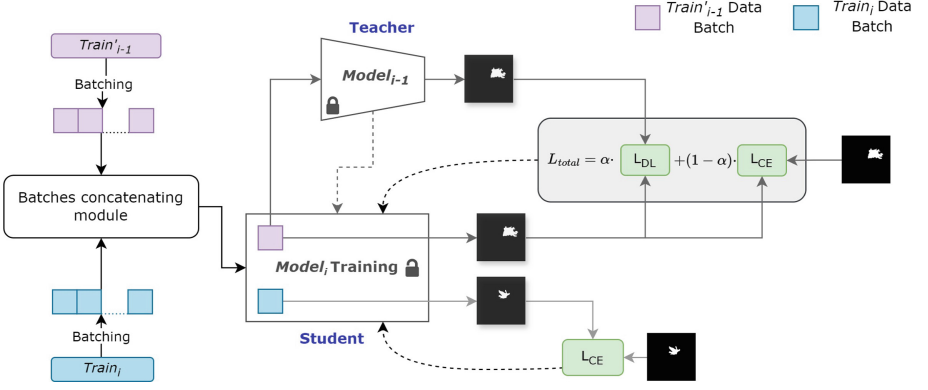


Fig. 3. Our proposed Recall-Based Distillation approach begins with separate batching of images from $Train'_{i-1}$ (subset of previous task data $Train_{i-1}$) and $Train_i$ (current task data) in the Batches concatenating module. Next, a Distillation Loss strategy is applied to recall knowledge from the previous model ($Model_{i-1}$), optimizing both losses to facilitate learning across new and old data distributions.

separately, it may lead to bias towards the one presented first, potentially limiting the model’s adaptability. A more systematic approach is to alternate batches from the two distributions. By interweaving data from different distributions, we hypothesize that the model learns to adapt to both sets simultaneously, leading to more balanced and efficient learning.

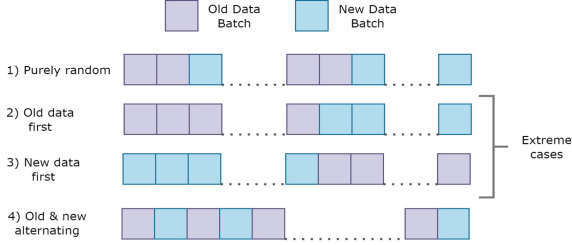


Fig. 4. Four curriculum’s for concatenating batches from $Train'_{i-1}$ and $Train_i$.

Distillation Loss: After the batches from the two distributions have been concatenated in the batch concatenating module, the $Model_i$ is trained using the weights initialized from $Model_{i-1}$. During training, batches originating from $Train_i$ are trained on the cross-entropy loss L_{CE} . To recall knowledge of the old model $Model_{i-1}$ a distillation loss is applied on $Train'_{i-1}$ batches, given as:

$$L_{DL} = \frac{1}{N} \sum_{i \in N} KL(p_i^s || p_i^t). \quad (2)$$

$KL(\cdot)$ is the Kullback-Leibler divergence function. p_i^s and p_i^t represent the probabilities of the i^{th} pixel in the segmentation map extracted from the student and the teacher network respectively, and N is total number of pixels in the map. Hence the total loss optimized for $Train'_{i-1}$ batches is a combination of L_{CE} and L_{DL}

$$L_{total} = (1 - \alpha) * L_{CE} + \alpha * L_{DL} \quad (3)$$

where α is a hyper-parameter that denotes the weights of each term. Distillation loss is exclusively applied to the old data $Train'_{i-1}$ and not the new data $Train_i$, as the old model $Model_{i-1}$ has only been exposed to the old data. Therefore, we distill the knowledge from the old data onto the new model $Model_i$.

4 Experiments

4.1 Implementation Details

We conduct experiments using the U-net architecture [29], which comprises an encoder-decoder structure. The encoder extracts high-level features through convolutions and max-pooling, while the decoder enhances spatial dimensions using transposed convolutions. Skip connections preserve detailed information. For training the U-net models, we employed the Adam optimizer [17]. The batch size was set to 32 and the models were trained for 30 epochs, during which the learning rate was fixed at 0.001. Additionally, we utilized a pacing function, specifically the ReduceOnPlateau scheduler, to dynamically adjust the learning rate during training. This pacing function reduces the learning rate when the validation loss plateaus, allowing the model to converge more efficiently.

4.2 Datasets

We apply our approach in the context of two application domains: 1) top salt segmentation in seismic images and 2) brain tumor segmentation in magnetic resonance images (MRIs).

Salt Segmentation in Seismic: As shown in Table 1, we used 3 distinct seismic datasets from the coast of Gulf of Mexico (GOM) to train and evaluate our models. GOM_A , GOM_B , and GOM_C , are 3D seismic volumes from which 2D images, measuring 256×256 pixels are sliced at regular intervals in an inline and crossline manner. GOM_A is a substantial volume with $14k$ images. GOM_B and GOM_C are comparatively smaller in scale with nearly $4k$ images each. At each increment a non-intersecting set of these datasets is used as $Test^s$. For $Test^u$, we used two separate 3D volume, also from GOM, denoted by $Test_1^u$ and $Test_2^u$.

Table 1. Distribution of Train and Test Seismic &MRI Datasets

Domain	Seismic		MRI	
	Dataset	Data size	Dataset	Data size
Train	GOM_A	14,000	UCSF	6,148
	GOM_B	4,000	UPENN	4,215
	GOM_C	4,000	TCGA	2,128
Test-seen	$GOM'_A + GOM'_B$	500	UCSF' + UPENN'	500
	GOM'_C	500	TCGA'	300
Test-unseen	$Test_1^u$	450	BraTS	3,475
	$Test_2^u$	500		

Brain Tumor Segmentation in MRI: As shown in Table 1, we utilized four MRI datasets: UPENN [4], UCSF [7], BraTS20 [2, 3, 23], and TCGA [6, 21]. UCSF includes 3D MRI images gathered from the University of California, San Francisco and includes nearly 6k 2D generated images after slicing the 3D scans. UPENN comprises approximately 4.2k 2D generated images, with each image typically having dimensions of 256×256 pixels. BraTS is a widely-used benchmark dataset consisting of MRI images collected from multiple institutions and we used approximately 3.4k 2D images. The TCGA-TCIA dataset combines MRI images from The Cancer Genome Atlas (TCGA) collection and from The Cancer Imaging Archive from which we used nearly 2k MRI images.

4.3 Evaluation Metrics

For model evaluation, we employ three primary metrics for model evaluation: the dice coefficient, the Area Under the Curve (AUC) of the Precision-Recall (PR) score, and the confusion matrix. The dice coefficient quantifies the degree of overlap between the binary segmentation mask predicted by our model and the ground truth, providing a measure of segmentation accuracy. The AUC of the PR curve offers a comprehensive assessment of binary classification performance by illustrating the trade-off between precision and recall across various threshold values. Further, a pixel-wise confusion matrix, is incorporated to further analyze the model’s performance. It consists of four key elements:

- True Positives (TP): Correctly predicting the positive class
- False Positives (FP): Mistakenly identifying positive class where none exists.
- True Negatives (TN): Accurately predicting the negative class.
- False Negatives (FN): Mistakenly identifying negative class where none exists.

5 Results and Discussion

5.1 Recall-Based Knowledge Distillation on Incremental Learning

Table 2. Summary of models trained using recall-based knowledge distillation. The subscript represents the number of images from the respective dataset.

Model	Train data	Abbreviation	Training Time
Teacher	$(\text{GOM}_A + \text{GOM}_B)_{19k}$	<i>Teacher</i>	21478.26 s
Baseline [19]	$(\text{GOM}_A + \text{GOM}_B)_{19k} + (\text{GOM}_C)_{4k}$	<i>Baseline</i>	4521.74 s
Model retraining [33]	$(\text{GOM}_A + \text{GOM}_B + \text{GOM}_C)_{23k}$	<i>Retrained</i>	26015.37 s
Recall-Based KD (our)	$(\text{GOM}_A + \text{GOM}_B)_{2k} + (\text{GOM}_C)_{4k}$	<i>Student_{2k}</i>	6782.6 s
	$(\text{GOM}_A + \text{GOM}_B)_{4k} + (\text{GOM}_C)_{4k}$	<i>Student_{4k}</i>	9043.47 s
	$(\text{GOM}_A + \text{GOM}_B)_{6k} + (\text{GOM}_C)_{4k}$	<i>Student_{6k}</i>	11304.34 s

Table 3. Results of recall-based KD approach on Seismic using curriculum 1 in Fig. 4.

Test Data	Metric	<i>Teacher</i>	<i>Baseline</i> [19]	<i>Retrained</i> [33]	KD approach (our)		
					<i>Student_{2k}</i>	<i>Student_{4k}</i>	<i>Student_{6k}</i>
$(\text{GOM}_A + \text{GOM}_B)_{500}$	Dice	0.93778	0.15217	0.93136	0.86637	0.91597	0.91349
	AUC PR	0.88807	0.1142	0.87783	0.80333	0.86942	0.86738
	TP	0.9828	0.11044	0.98146	0.82798	0.89853	0.89727
	FP	0.10079	0.38373	0.10874	0.04646	0.04225	0.04241
	FN	0.01719	0.8895	0.01853	0.17202	0.10146	0.10273
	TN	0.91707	0.95427	0.91669	0.92515	0.92248	0.92243
$(\text{GOM}_C)_{500}$	Dice	0.23052	0.889514	0.86394	0.864523	0.857949	0.86151
	AUC PR	0.25158	0.78457	0.48332	0.73805	0.73661	0.75105
	TP	0.4097	0.93266	0.96943	0.76075	0.7596	0.77544
	FP	0.80123	0.22347	0.26859	0.07001	0.08268	0.08928
	FN	0.5903	0.06733	0.03056	0.23924	0.24039	0.22455
	TN	0.91573	0.93164	0.93093	0.9401	0.94023	0.93963
Test ₁ ^u	Dice	0.482464	0.434251	0.41277	0.476423	0.4639002	0.4623619
	AUC PR	0.36989	0.32812	0.32478	0.33958	0.3423	0.32814
	TP	0.52588	0.52918	0.53436	0.52626	0.49709	0.47291
	FP	0.57412	0.62335	0.66374	0.56479	0.55718	0.54772
	FN	0.47412	0.4808	0.46564	0.47374	0.5129	0.52709
	TN	0.95724	0.95965	0.95607	0.96252	0.96437	0.96529
Test ₂ ^u	Dice	0.51567	0.21028	0.52429	0.34474	0.41276	0.41102
	AUC PR	0.40385	0.16906	0.43402	0.27965	0.3375	0.30232
	TP	0.51223	0.1742	0.53412	0.27293	0.33873	0.32908
	FP	0.43038	0.36876	0.38271	0.19647	0.19748	0.19811
	FN	0.48776	0.82579	0.46588	0.72707	0.66126	0.68292
	TN	0.93169	0.95365	0.93236	0.94997	0.9467	0.95002

Table 4. Different models trained using MRI datasets.

Model	Train data	Abbreviation	Training Time
Teacher	$(\text{UPENN} + \text{UCSF})_{12k}$	<i>Teacher</i>	1868 s
Baseline [19]	$(\text{UPENN} + \text{UCSF})_{12k} + (\text{TCGA})_{2k}$	<i>Baseline</i>	348 s
Model retraining [33]	$(\text{UPENN} + \text{UCSF} + \text{TCGA})_{14k}$	<i>Retrained</i>	2280 s
Recall-Based KD	$(\text{UPENN} + \text{UCSF})_{1k} + (\text{TCGA})_{2k}$	<i>Student_{1k}</i>	832 s
	$(\text{UPENN} + \text{UCSF})_{2k} + (\text{TCGA})_{2k}$	<i>Student_{2k}</i>	910 s
	$(\text{UPENN} + \text{UCSF})_{3k} + (\text{TCGA})_{2k}$	<i>Student_{3k}</i>	1003 s

Seismic Data: As shown in Table 2, we begin by taking model trained with $\text{GOM}_A + \text{GOM}_B$ datasets as the teacher model. Then starting with the weights of *teacher* we incrementally train a model with GOM_C dataset which acts as a baseline model, a state of the art [19], for comparing our approach. Now, we train multiple models using our recall-based distillation approach by varying the number of images to use for the recall process. For instance, *Student_{2k}* comprises a model trained with GOM_C dataset and $2k$ images from the old datasets $\text{GOM}_A + \text{GOM}_B$. Table 3 details the results obtained on these model. $(\text{GOM}_A + \text{GOM}_B)_{500}$ and $(\text{GOM}_C)_{500}$ indicates testing on a portion of seen datasets while Test_1^u and Test_2^u are unseen test datasets. Additionally, we have trained the only state-of-the-art method, to the best of our knowledge, by using all three datasets together, thus retraining the model from scratch [33]. The training time for each model highlights that the retrained model requires the most time since all the old data is reintroduced for training. In contrast, our method demonstrates that using KD on a portion of the old data results in significantly faster training, thereby reducing computational complexity and training time (Table 2).

Looking at the results in Table 3, some general trends can be observed. First, the teacher model, which initially seemed to be performing well on testing on $(\text{GOM}_A + \text{GOM}_B)_{500}$ dataset, shows a drastic fall in performance on the baseline model. However, applying the recall-based KD approach seems to work in retaining the original performance on this dataset as this is the set of images on which distillation was applied. We see a drop in performance on unseen datasets as well when tested with the baseline model compared to the initial teacher. But as recall-based KD approach is applied the model once again retains the performance. Further, as the number of images from the old datasets $(\text{GOM}_A + \text{GOM}_B)$ is increased, we generally see a slight improvement in performance over seen datasets.

MRI Data: For evaluation on MRI datasets, teacher model is trained on $(\text{UPENN} + \text{UCSF})_{12k}$ datasets. The baseline model is then trained incrementally with the addition of $(\text{TCGA})_{2k}$ dataset, starting with weights of previous model $(\text{UPENN} + \text{UCSF})_{12k}$. Subsequently, our recall-based KD approach trains multiple models with varying numbers of images from the old datasets

Table 5. Results of recall-based KD approach on MRI using curriculum 1 in Fig. 4.

Test Data	Metric	Teacher	Baseline [19]	Retrained [33]	KD approach (our)		
					$Student_{1k}$	$Student_{2k}$	$Student_{3k}$
(UPENN + UCSF) ₅₀₀	Dice	0.79622	0.31561	0.78667	0.60231	0.62445	0.74687
	AUC PR	0.89794	0.51766	0.79888	0.79911	0.86853	0.87065
	TP	0.80639	0.25993	0.69232	0.57186	0.55377	0.74967
	FP	0.12985	0.06911	0.09664	0.07853	0.03944	0.12908
	FN	0.19361	0.96702	0.43922	0.42414	0.44223	0.24633
	TN	0.95232	0.74006	0.95644	0.95776	0.95924	0.95316
(TCGA) ₅₀₀	Dice	0.10646	0.34142	0.33211	0.33822	0.32351	0.32001
	AUC PR	0.20899	0.38868	0.37865	0.38561	0.38796	0.37489
	TP	0.21527	0.40554	0.33902	0.35174	0.32187	0.33711
	FP	0.53768	0.18782	0.13294	0.13345	0.07438	0.14749
	FN	0.78472	0.19446	0.08934	0.08825	0.11813	0.10289
	TN	0.97905	0.97103	0.97543	0.97115	0.97289	0.97122
BraTS	Dice	0.73705	0.38182	0.57844	0.59197	0.54139	0.69398
	AUC PR	0.86031	0.5788	0.73485	0.78705	0.79967	0.82352
	TP	0.73297	0.32657	0.54333	0.55996	0.47719	0.69844
	FP	0.12969	0.11271	0.94545	0.09776	0.04192	0.14609
	FN	0.26511	0.67343	0.34089	0.44003	0.51733	0.29609
	TN	0.94849	0.96178	0.95098	0.95482	0.95821	0.94998

Table 6. Analyzing the performance using different training curriculum’s on the batches for Seismic Models.

Test Data	Metric	$Student_{2k}$ Model			
		C 1	C 2	C 3	C 4
(GOM _A + GOM _B) ₅₀₀	Dice	0.86637	0.51639	0.92185	0.87554
	AUC PR	0.80333	0.42904	0.87597	0.81443
(GOM _C) ₅₀₀	Dice	0.864523	0.87385	0.80539	0.74283
	AUC PR	0.73805	0.74559	0.72343	0.56166
Test ₁ ^u	Dice	0.4764236	0.46001	0.46646	0.49214
	AUC PR	0.32407	0.32342	0.30921	0.33958
Test ₂ ^u	Dice	0.34474	0.20400	0.36321	0.38687
	AUC PR	0.27965	0.17053	0.30587	0.31769

(UPENN + UCSF) combined with (TCGA)_{2k} dataset. These models are denoted as $Student_{1k}$, $Student_{2k}$, and $Student_{3k}$, indicating the number of images from the old datasets used for training. Here as well, we retrained a state-of-the-art model from scratch using all three datasets [33], which took the longest time due to reintroducing all old data. In contrast, our KD method uses a portion of old data, resulting in significantly faster training and reduced computational complexity.

For MRI datasets, the results are summarized in Table 5. (UPENN + UCSF)₅₀₀ and (TCGA)₅₀₀ indicates testing on a portion of seen datasets while BraTS is an unseen test dataset. Notably, the teacher model (*Teacher*) performs well initially, but there is a decrease in performance on the baseline model. However, employing the recall-based KD approach with varying combinations of old and new datasets shows promising results in maintaining or even improving performance metrics. For instance, the *Student*_{3k} model shows improved performance compared to the baseline in several metrics across different test datasets. This trend continues as the number of images from the old dataset increases, indicating the effectiveness of the recall-based KD approach in incremental learning scenarios. Some sample predictions made by these models can be seen in Fig. 5.

5.2 Further Evaluation

Applying Learning Curriculum on Batches: We investigate the effects of using different curriculum’s as depicted in Fig. 4. Table 6 shows the results for salt segmentation in Seismic data and Table 7 shows the results for tumor segmentation in MRI data. C1 represents the curriculum where batches from both new and old datasets are given purely randomly. C2 and C3 represent the extreme cases where the batches from one dataset are given first, followed by batches from the other dataset. C4 is the balanced case where batches from each set are given alternately.

Interestingly, when analyzing the results on the seen datasets in MRI, we see that when using curriculum C2, where batches from TCGA are given first, followed by batches from UPENN+UCSF, the model performs best in making predictions on UPENN+UCSF test dataset. Similar observation can be seen when using C3 where the model performs best on TCGA test dataset as the batches from TCGA are given for training last. This observation aligns with the intuition that when the model is exposed to batches from UPENN+UCSF or TCGA towards the end of training, it tends to perform better on the respective test sets. Conversely, on the unseen dataset BraTS, the best performance is achieved when batches from the two datasets are alternated (as in C4).

In case of seismic we observe a similar performance. When using C2 or C3 the model excels in making predictions on the respective test datasets correlated with the dataset introduced towards the end of training. Further, on unseen test datasets the model trained with alternating batches curriculum (C4) tends to do best. This suggests that providing a balanced exposure to both datasets during training may lead to improved generalization and performance on unseen data. Hence, it is essential to carefully designing the training curriculum to optimize model performance across different datasets and tasks. By strategically alternating between the dataset distributions during training, models can effectively leverage the diverse information present in each dataset, ultimately enhancing their ability to generalize to new and unseen data distributions.

Varying Loss Hyperparameter: Here we investigate the choice for the hyperparameter α in the loss function. We train with different values of α ranging from

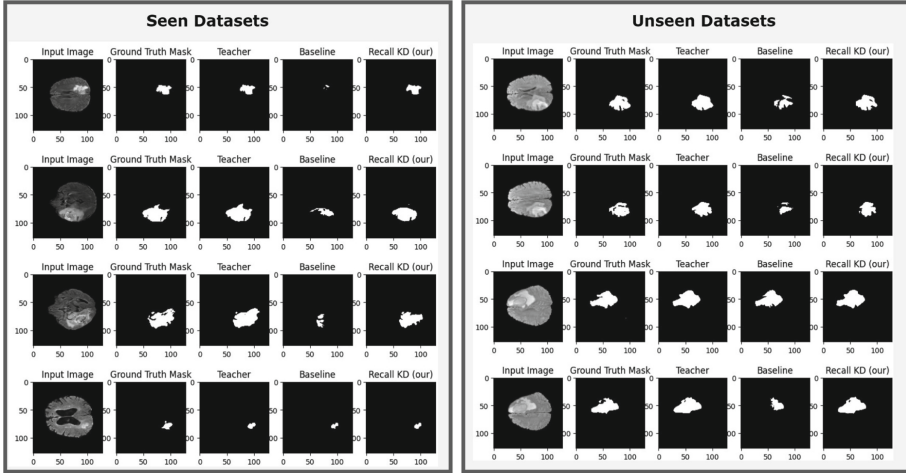


Fig. 5. Qualitative results on MRI seen and unseen datasets.

Table 7. Analyzing the performance using different training curriculum’s on the batches for MRI models.

Test Data	Metric	$1k_{old} + 2k_{new}$ Model			
		C 1	C 2	C 3	C 4
(UPENN + UCSF) ₅₀₀	Dice	0.64401	0.553229	0.730802	0.681149
	AUC PR	0.80855	0.781559	0.847799	0.832356
(TCGA) ₅₀₀	Dice	0.33822	0.340035	0.311825	0.334421
	AUC PR	0.38561	0.387077	0.362951	0.391385
BraTS	Dice	0.59197	0.376764	0.600653	0.605210
	AUC PR	0.78705	0.676064	0.777815	0.788173

0 to 1 and observe their impact on model performance. This results are shown in Table 8 and Table 9 for Seismic and MRI respectively. In each case, in seen and unseen dataset testing, we see that the model performs best with an α value of 0.4. In scenarios where only distillation loss is utilized (i.e., $\alpha = 1$), the model focuses excessively on mimicking the outputs of the teacher model without fully leveraging the information contained in the ground truth labels. This can potentially lead to a loss of fine-grained details and an over-reliance on the teacher model’s predictions. On the other hand, when $\alpha = 0.4$, a balance is struck between distillation loss and cross-entropy loss. This allows the model to not only benefit from the distilled knowledge of the teacher model but also to refine its predictions based on the ground truth labels, thus leading to improved generalization and robustness.

Table 8. Analyzing the effect of hyperparameter α , ranging from 0 to 1, in computing loss for Seismic models. Lower values of α indicate higher emphasis on cross-entropy loss, while higher values prioritize distillation loss.

Test Data	Metric	Recall-based KD Model			
		$\alpha = 0$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 1.0$
$(\text{GOM}_A + \text{GOM}_B)_{500}$	Dice	0.85432	0.86637	0.86362	0.75942
	AUC PR	0.77453	0.80333	0.79324	0.64341
$(\text{GOM}_C)_{500}$	Dice	0.86123	0.86452	0.85647	0.86237
	AUC PR	0.73281	0.73805	0.73222	0.72311
Test ₁ ^u	Dice	0.44839	0.47642	0.46211	0.41903
	AUC PR	0.30123	0.33958	0.31955	0.29585
Test ₂ ^u	Dice	0.31309	0.34474	0.32567	0.29053
	AUC PR	0.25849	0.27965	0.27001	0.24473

Table 9. Effect of hyperparameter α in computing loss for MRI models. In each test set, choosing α of 0.4 seemed to give the optimal balance

Test Data	Metric	Recall-based KD Model			
		$\alpha = 0$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 1.0$
$(\text{UPENN} + \text{UCSF})_{500}$	Dice	0.668466	0.681149	0.675805	0.565161
	AUC PR	0.813678	0.832356	0.827396	0.707104
$(\text{TCGA})_{500}$	Dice	0.332678	0.334421	0.324947	0.32346
	AUC PR	0.388366	0.391385	0.391299	0.385238
BraTS	Dice	0.57996	0.605210	0.601866	0.523672
	AUC PR	0.76794	0.788173	0.770234	0.635562

6 Conclusion

In this paper, we propose a novel approach to mitigate catastrophic forgetting in incremental semantic segmentation, focusing on distribution-based forgetting. Our recall-based distillation method combines knowledge distillation and recall learning techniques to enhance model robustness across dynamic environments. By reintroducing a small portion of previous data during training and employing tailored distillation, our approach improves model stability and performance. Through extensive evaluations on seismic and MRI datasets, we demonstrate its effectiveness in maintaining performance across varying data distributions. Thus, our method preserves information transfer by periodically reintroducing past data during training, reinforcing and integrating earlier knowledge to prevent catastrophic forgetting. Future work will involve empirical verification of this information preservation aspect, where we will conduct detailed studies to measure the extent to which knowledge from earlier steps is retained

and utilized in subsequent training phases. This will further strengthen the reliability and applicability of our proposed method in real-world scenarios.

Acknowledgements. We would like to thank Shell Global Solutions for providing us with the Gulf of Mexico Seismic datasets.

References

1. Aslam, N., Khan, I.U., Bashamakh, A., Alghool, F.A., et al.: Multiple sclerosis diagnosis using machine learning and deep learning: challenges and opportunities. *Sensors* (2022)
2. Bakas, S., Akbari, H., Sotiras, A., Bilello, M., Rozycki, M., et al.: Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features. *Sci. Data* (2017)
3. Bakas, S., Reyes, M., Jakab, A., Bauer, S., Rempfler, M., et al.: Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge. *arXiv preprint* (2018)
4. Bakas, S., Sako, C., Akbari, H., Bilello, M., et al.: The university of Pennsylvania glioblastoma (UPenn-GBM) cohort: advanced MRI, clinical, genomics, & radiomics. *Sci. Data* (2022)
5. Beyer, L., Zhai, X., Royer, A., Markeeva, L., et al.: Knowledge distillation: a good teacher is patient and consistent. In: *CVPR* (2022)
6. Buda, M., Saha, A., Mazurowski, M.A.: Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm. *Comput. Biol. Med.* (2019)
7. Calabrese, E., Villanueva-Meyer, J.E., Rudie, J.D., Rauschecker, A.M., et al.: The University of California San Francisco Preoperative Diffuse Glioma MRI dataset. *Radiology: Artificial Intelligence* (2022)
8. Cermelli, F., Mancini, M., Bulò, S.R., Ricci, E., Caputo, B.: Modeling the background for incremental learning in semantic segmentation. In: *CVPR* (2020)
9. Devarakota, P., Gala, A., Li, Z., Alkan, E., Cai, Y., et al.: Deep learning in salt interpretation from R&D to deployment: challenges and lessons learned. In: *Second International Meeting for Applied Geoscience & Energy* (2022)
10. Douillard, A., Chen, Y., Dapogny, A., Cord, M.: Plop: learning without forgetting for continual semantic segmentation. In: *CVPR* (2021)
11. Goodfellow, I.J., Mirza, M., Xiao, D., Courville, A., Bengio, Y.: An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint* (2013)
12. Gu, Y., Deng, C., Wei, K.: Class-incremental instance segmentation via multi-teacher networks. In: *AAAI* (2021)
13. Hao, S., Zhou, Y., Guo, Y.: A brief survey on semantic segmentation with deep learning. *Neurocomputing* (2020)
14. Hassan, T., Shafay, M., Hassan, B., Akram, M.U., et al.: Knowledge distillation driven instance segmentation for grading prostate cancer. *Comput. Biol. Med.* (2022)
15. Huang, Z., Hao, W., Wang, X., Tao, M., et al.: Half-real half-fake distillation for class-incremental semantic segmentation. *arXiv preprint* (2021)
16. Khan, I.U., Aslam, N., Anis, F.M., Mirza, S., et al.: Amniotic fluid classification and artificial intelligence: challenges and opportunities. *Sensors* (2022)

17. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint (2014)
18. Lateef, F., Ruichek, Y.: Survey on semantic segmentation using deep learning techniques. *Neurocomputing* (2019)
19. Li, Z., Hoiem, D.: Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(12), 2935–2947 (2017)
20. Maracani, A., Michieli, U., Toldo, M., Zanuttigh, P.: Recall: replay-based continual learning in semantic segmentation. In: *CVPR* (2021)
21. Mazurowski, M.A., Clark, K., Czarnek, N.M., Shamsesfandabadi, P., et al.: Radiogenomics of lower-grade glioma: algorithmically-assessed tumor shape is associated with tumor genomic subtypes and patient outcomes in a multi-institutional study with the cancer genome atlas data. *J. Neuro-oncology* (2017)
22. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: the sequential learning problem. In: *Psychology of Learning and Motivation* (1989)
23. Menze, B.H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., et al.: The multimodal brain tumor image segmentation benchmark (brats). *IEEE Trans. Med. Imaging* (2014)
24. Michieli, U., Zanuttigh, P.: Incremental learning techniques for semantic segmentation. In: *CVPR* (2019)
25. Michieli, U., Zanuttigh, P.: Knowledge distillation for incremental learning in semantic segmentation. *Comput. Vis. Image Understanding* (2021)
26. Mirza, S., Nguyen, V.D., Mantini, P., Shah, S.K.: Data quality aware approaches for addressing model drift of semantic segmentation models. In: *VISIGRAPP (3: VISAPP)* (2024)
27. Phan, M.H., Phung, S.L., Tran-Thanh, L., Bouzerdoum, A., et al.: Class similarity weighted knowledge distillation for continual semantic segmentation. In: *CVPR* (2022)
28. Qu, H., Rahmani, H., Xu, L., Williams, B., Liu, J.: Recent advances of continual learning in computer vision: an overview. arXiv (2021)
29. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: *MICCAI* (2015)
30. Shang, C., Li, H., Meng, F., Wu, Q., et al.: Incrementer: transformer for class-incremental semantic segmentation with knowledge distillation focusing on old class. In: *CVPR* (2023)
31. Yan, S., Zhou, J., Xie, J., Zhang, S., He, X.: An EM framework for online incremental learning of semantic segmentation. In: *ACMMM* (2021)
32. Yuan, B., Zhao, D.: A survey on continual semantic segmentation: theory, challenge, method and application. arXiv preprint (2023)
33. Zhang, Y., Yang, Q.: An overview of multi-task learning. *Natl. Sci. Rev.* **5**(1), 30–43 (2018)



Contrastive Gaussian Clustering for Weakly Supervised 3D Scene Segmentation

Myrna Castillo^(✉), Mahtab Dahaghin, Matteo Toso, and Alessio Del Bue

Pattern Analysis and Computer Vision (PAVIS), Istituto Italiano di Tecnologia (IIT),
Genoa, Italy

{myrna.castillo,mahtab.dahaghin,matteo.toso,alessiodel.bue}@iit.it

Abstract. 3D scene segmentation is a crucial task in Computer Vision, with applications in autonomous driving, augmented reality, and robotics. Traditional methods often struggle to provide consistent and accurate segmentation across different viewpoints. To address this, we look at the growing field of novel view synthesis. Methods like NeRF and 3DGS take a set of images and implicitly learn a multi-view consistent representation of the geometry of the scene; the same strategy can be extended to learn a 3D segmentation of the scene that is consistent with the 2D segmentation of an initial training set of input images.

We introduce *Contrastive Gaussian Clustering*, a novel approach for novel segmentation view synthesis and 3D scene segmentation. We extend 3D Gaussian Splatting to include a learnable 3D feature field, which allows us to cluster the 3D Gaussians into objects. Using a combination of contrastive learning and spatial regularization, our model can be trained on inconsistent 2D segmentation labels, and still learn to generate multi-view consistent masks. Moreover, the resulting model is extremely accurate, improving the IoU accuracy of the predicted masks by +8% over the state of the art.

Code and trained models are available at <https://github.com/MyrnaCCS/contrastive-gaussian-clustering>.

Keywords: 3D Gaussian Splatting · 3D Segmentation · Contrastive Learning

1 Introduction

Reliable and efficient 3D scene segmentation, *i.e.*, the ability to divide the content of a 3D scene into different objects, is a fundamental skill at the core of several

M. Castillo and M. Dahaghin—These authors contributed equally to this work.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-78347-0_8.

© The Author(s) 2025

A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15323, pp. 114–130, 2025.

https://doi.org/10.1007/978-3-031-78347-0_8

computer vision tasks, and it is a prerequisite for autonomous navigation, scene understanding, and for many AR/VR applications [13]. In this work, we propose a general 3D scene segmentation approach based on 3D Gaussian Splatting [16], that only requires 2D images and their segmentation masks as input, without making assumptions on the masks’ consistency across images.

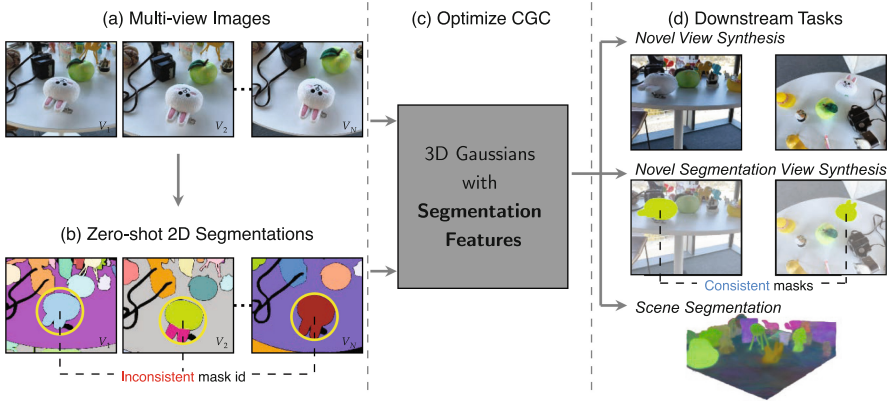


Fig. 1. The objective of *Contrastive Gaussian Clustering* is to take (a) a set of input images and (b) their class-agnostic segmentations and (c) distill their information in a model based on 3DGS. This model can then be used for (d) a wide range of visual and segmentation downstream tasks, such as novel view synthesis, retrieving the mask of a selected object, or 3D scene segmentation.

One of the challenges of 3D scene segmentation is the limited availability of annotated 3D scene datasets, as manual annotations are time-consuming [15]. Recent works bypassed this issue by lifting readily available 2D image understanding to 3D space [28], inserting their semantic information into 3D point clouds [14, 29, 30] or NeRFs [17, 19, 40]. These methods have shown that averaging noisy labels across multiple views generates view-independent dense semantic labels [40]. Early approaches relied on a limited range of task-specific labels [7, 36], but the recent introduction of foundational models like CLIP [32] and SAM [18] provide open-vocabulary 2D semantic segmentation labels, which can be used to optimize scene representations [31, 37]. The segmentation masks generated by the foundation models, however, are not always consistent across views, and existing methods require time-consuming pre-processing to enforce cross-view consistency in the training data [37]. In this work, we address this by introducing a model that can be trained on inconsistent 2D segmentation masks, while still learning a 3D feature field consistent across all views.

As exemplified in Fig. 1, our method takes as input a) a set of multi-view images and b) their 2D segmentations, which are not required to be consistent across views. We then use images and masks to train c) a model representing both the visual and geometrical information of the scene, as well as a 3D segmentation feature field. This model can then be used for a wide range of d)

downstream tasks exploiting the visual information (novel view synthesis), segmentation information (3D scene segmentation) or on a combination of the two (returning a segmentation mask given a selected point on a rendered view). The optimization of the geometric and visual components can be approached following standard 3D Gaussian Splatting [16], using a rendering loss to optimize the color, position and shape of the 3D Gaussians. To learn the 3D segmentation feature field, we propose extracting information from the inconsistent 2D segmentation masks via contrastive learning. This approach ensures segmentation consistency across all views without requiring changes to the 2D masks themselves. We test the proposed method against related works based on implicit scene representations [17] and 3D Gaussian representations [31,37], and show through qualitative and quantitative evaluation how our method can match and outperform them. A video outlining the motivation and main results of this paper is available at the project’s page. Our contributions can be summarized as:

- A novel approach to embed a *3D feature field* in a 3DGS model, enabling simultaneous modelling of the scene’s appearance and segmentation.
- A contrastive-learning approach enforcing a multi-view consistent segmentation feature field, even when training on inconsistent segmentation masks.
- An approach for 3D scene segmentation by clustering the Gaussians according to the feature field.

2 Related Work

In this section, we provide an overview of the relevant literature on image and scene segmentation, in addition to 3D scene modeling with techniques for novel-view synthesis. For a complete review of scene understanding or semantic segmentation, we refer the reader to [27] and [11], respectively.

Scene Understanding. Scene understanding is a fundamental problem in computer vision, inferring the semantics and properties of all elements in a 3D scene given a 3D model and a set of RGB images [28]. Early approaches train models on ground-truth (GT) 3D labels, focusing on specific tasks like 3D object classification [36], object detection and localization [7] or 3D semantic and instance segmentation [2,6,9,21]. To overcome the limited availability of 3D GT data, subsequent work leverages 2D supervision, by back-projecting and fusing 2D labels to generate pseudo 3D annotations [12] or applying contrastive learning between 2D and 3D features [24,33]. More recently, large visual language models [4,18,32] have allowed to shift from a close-set of predefined labels to an open-vocabulary framework [32], making it possible zero-shot transfer to new tasks and dataset distributions. We also leverage contrastive learning and foundation models, using class-agnostic segmentation masks generated by the Segment Anything Model (SAM) [18]. However, we apply such techniques to a different scene representation - 3D Gaussian Splatting - and combine contrastive loss with other forms of supervision, like spatial regularization from the distance between the Gaussians.

Radiance Fields. Neural Radiance Fields (NeRF) [25] optimize a Multilayer Perceptron (MLP) to represent a 3D scene as a continuous volumetric function that maps position and viewing direction to density and color. NeRF has enabled the rendering of complex scenes, producing high-quality results in novel-view synthesis. Subsequent work have focused on faster training/rendering [1, 26, 39]. An alternative approach to NVS comes from 3D Gaussian Splatting (3DGS) [16], which achieves both competitive training times and real-time rendering at higher image resolution. Unlike NeRF [25], 3DGS foregoes a continuous volumetric representation and instead approximate a scene using millions of 3D Gaussians with different sizes, orientations, and view-dependent colors. One of the advantages of this approach is that it allows for direct access to the radiance field data, enabling to edit the scene by removing, displacing or adding Gaussians [8]. This also allows capturing dynamic scenes, including a time parameter to model the scene’s changes over time [35]; or combining the model in a pipeline with a foundation model to edit the scene from text prompts [10] or select the Gaussians associated with a specific object [37]. Of these methods, Gaussian Grouping is the closest to our application by segmenting the scene into groups of 3D Gaussians. However, this technique relies on a video-tracker to obtain consistent masks IDs across the training images, which also preset the number of instances in the scene.

Scene Understanding in Radiance Fields Representations. Semantic-NeRF [40] extends the implicit scene representation to encode appearance, geometry, and semantics, and generates denoised semantic labels by training over sparse or noisy annotations. Other methods propose to distill image embeddings extracted by a foundation model encoder into a 3D feature field. Distilled Feature Fields (DFF) [34] includes an extra branch that outputs a pixel-aligned feature vector extracted from LSeg [20] or DINO [4]. Unlike DFF, LERF [17] supervises by rendering non pixel-aligned multi-scale CLIP [32] embeddings. Although these techniques locate a wide variety of objects given any language prompt, they may suffer from inaccurate segmentations occasionally caused by objects with similar semantics. More recent methods [5, 31] have used foundational models for grounding language/segmentation features onto the 3D Gaussians. While these methods provide better performance in localization tasks, achieving higher accuracy, their segmentation masks are noisy/patchy. Mingqiao *et al.* [37] cluster the Gaussians by assigning them a unique identity ID. Though these methods can include instance segmentation features into the scene representation, the number of objects in the scene is predefined, and it requires an additional tracking method to pre-compute the needed multi-view consistent segmentation labels. A similar approach of using contrastive learning to lift inconsistent 2D segmentations into NeRF has also been used in 3D instance segmentation [3]. We show an alternative method to encode identity features into 3D Gaussians, so we can group them into clusters that we can easily extract/remove from the 3D scene.

3 Methodology

In this work, we represent a scene as a collection of 3D Gaussians that jointly model geometry, appearance, and instance segmentation information. Our app-

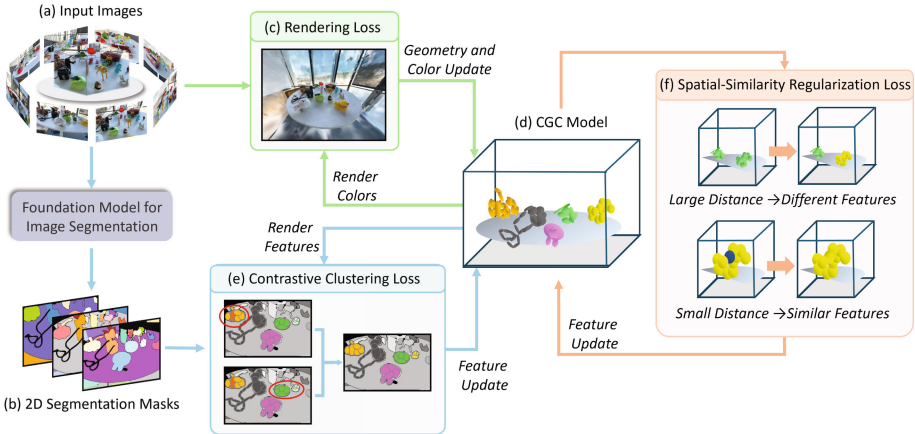


Fig. 2. Pipeline: (a) Given a set of images from different viewpoints, we use (b) a foundation model for image segmentation to generate 2D segmentation masks. We capture the appearance of the scene using (c) a rendering loss that, like in traditional 3DGS, optimizes the geometry and color of (d) our *Contrastive Gaussian Clustering* model. Simultaneously, (e) a contrastive clustering loss on the rendered-features optimizes a 3D segmentation feature field, encoded in (d) our scene model; this loss pulls apart the rendered-features of pixels belonging to different masks and encourages similarity between the features of those belonging to the same mask. Moreover, we use (f) a spatial-similarity regularization mechanism, encouraging the segmentation features to be similar for neighboring Gaussians and different for faraway Gaussians. (Color figure online)

roach allows high-quality real-time novel view and segmentation synthesis. We empower a 3DGS model to tackle scene understanding downstream tasks by augmenting each 3D Gaussian with a view-independent feature vector. This set of learnable feature vectors is called the *3D feature field*. We optimize our 3D feature field to lift inconsistent 2D segmentation masks into 3D space. A post-optimization process is then applied to render multi-view consistent segmentations and to segment the scene into distinct clusters. A comparison between our algorithm and 3DGS is available in the Supplementary Material.

As shown in Fig. 2, our approach takes (a) a set of input images, from which we independently extract (b) inconsistent 2D segmentation masks using a foundation model for image segmentation. Then, we optimize the 3D Gaussians using (c) the original 3DGS loss function [16] that measures the difference between the rendered and ground truth images. Simultaneously, we make use of (e) a contrastive clustering loss to supervise the 3D feature field. This results in (d) a 3D Gaussian scene representation which captures both visual and instance information. To provide more accurate segmentations and speed up training, we introduce (f) a regularization term that enforces the correlation between the distance of Gaussians in Euclidean and the feature space.

In this section, we first review the 3DGS rendering method. Then we discuss the main steps of our pipeline, including rendering and supervising the 3D feature field via contrastive learning.

3.1 Preliminaries on 3D Gaussian Splatting

The 3DGS model represents a scene as millions of 3D Gaussians parameterized by their position μ , 3D covariance matrix Σ , opacity α , and color c . 3DGS represents the view-dependent appearance c by spherical harmonics. These parameters are jointly optimized to render high-quality novel-views. Since 3DGS preserves the properties of differentiable volumetric representations, it requires as input only a set of images and their camera parameters. Initially, 3DGS creates a set of 3D Gaussians using a sparse Structure-from-Motion (SfM) point cloud obtained during camera calibration. To render 3D Gaussians from a particular point of view, the 3DGS starts by projecting these Gaussians onto the image space, a rendering termed “splatting”. Subsequently, 3DGS generates a sorted list of \mathcal{N} Gaussians, ordering them from closest to farthest. The color of a pixel C is then computed by α -blending the colors of \mathcal{N} overlapping points:

$$C = \sum_{i \in \mathcal{N}} c_i \alpha'_i \prod_{j=1}^{i-1} (1 - \alpha'_j). \quad (1)$$

The final opacity α'_i is determined by multiplying the learned opacity α_i and the 2D Gaussian. The optimization is done by subsequent iterations that compare the ground-truth images against the corresponding rendered views.

3.2 3D Feature Field

The 3D feature field is a collection of learnable-vectors stored on the 3D Gaussians, that encode the instance segmentation of the scene. We augment each 3D Gaussian with a learnable feature f . Unlike the view-dependent appearance, this feature must remain consistent across all viewing directions. Therefore, instead of computing spherical harmonics coefficients, we extract its component from Gaussians. During training, we randomly initialize the feature vectors and then adjust them to minimize the contrastive clustering error. The optimization of the 3D feature field involves three iterative steps repeated for each training view: 3D feature field rendering; clustering the rendered features following the related GT segmentation map; and back-propagating the contrastive clustering error.

At each iteration, we render an image and its corresponding 2D feature map, following an analogous process to the rendering algorithm described in Sect. 3.1. For each pixel of the desired view, we α blend the features as:

$$F = \sum_{i \in \mathcal{N}} f_i \alpha'_i \prod_{j=1}^{i-1} (1 - \alpha'_j) \quad (2)$$

Contrastive Clustering. As the first step toward scene optimization, SAM automatically generates 2D segmentation masks from the set of input images. Specifically, we deploy SAM’s automatic generation pipeline on each training image $I \in \mathbb{R}^{H \times W}$, resulting in sets of segments $\{m^p \in \mathbb{R}^{H \times W} | p = 1 \dots \mathcal{N}_k\}$. The number of segments per image \mathcal{N}_k is uncapped, allowing the proposed approach to learn as many instances as are present in the scene. As described previously, the 3D feature field optimization is composed of three steps: rendering a 2D feature map for a given view, clustering the rendered features based on the corresponding 2D segmentation masks to compute a contrastive clustering loss, and then updating our 3D feature field accordingly.

Contrastive clustering maximizes the similarity among features within the same segment in the segmentation map, while minimizing it for those from different segments. Given a view p , the cluster $\{f^p\}$ is the set of rendered features of the 2D feature map that belongs to the same segment m^p in the corresponding GT segmentation map, and the mean feature in $\{f^p\}$ is the centroid \bar{f}^p . Like Contrastive Lift [3], we adopt a slow-fast contrastive learning strategy where the teacher parameters \bar{f}^p are updated by exponential moving average of the student parameters $\{f^p\}$. Our objective is to minimize the following loss function:

$$\mathcal{L}_{CC} = -\frac{1}{\mathcal{N}_k} \sum_{p=1}^{\mathcal{N}_k} \sum_{q=1}^{|\{f^p\}|} \log \frac{\exp(f_q^p \cdot \bar{f}^p / \phi^p)}{\sum_{s=1}^{\mathcal{N}_k} \exp(f_q^p \cdot \bar{f}^s / \phi^s)}, \quad (3)$$

where, f_q^p are features in $\{f^p\}$. The concentration estimation of the p -th cluster is ϕ^p . Similar to [38], we define it as: $\sum_{q=1}^{\mathcal{N}_p} \|f_q^p - \bar{f}^p\|_2 / \mathcal{N}_p \log(\mathcal{N}_p + \epsilon)$, where $\mathcal{N}_p = |\{f^p\}|$ and $\epsilon = 100$. ϕ is used to balance the cluster size and variance, and is small if the number of pixel-feature elements is high and the average distance between its elements and the centroid is small. The smooth parameter ϵ is needed to avoid excessively large ϕ . Rather than regularize the features by including a normalization loss, we apply ℓ_2 -normalization to each feature in the rendered feature map before the loss computation.

Spatial-Similarity Regularization. An easy way to obtain 3D instance segmentation is to cluster similar features. However, we occasionally observe sparse outliers (Gaussians misclassified) in regions where the scene is not well observed. Furthermore, we notice that constant failures in the 2D segmentation (*e.g.*, a chair that is inconsistently segmented in two parts: legs and seat) may induce to inaccurate segmentation masks.

To address these issues, we include spatial-similarity regularization to enforce spatial continuity of the feature vectors, encouraging adjacent 3D Gaussians to have similar segmentation feature vectors while discouraging faraway Gaussians from having the same segmentation features. The regularization function is computed with \mathcal{M} sampling Gaussians:

$$\mathcal{L}_{regularization} = \frac{\lambda_{near}}{\mathcal{M}\mathcal{K}} \sum_{j=1}^{\mathcal{M}} \sum_{i=1}^{\mathcal{K}} H(1 - f_j \cdot f_i) + \frac{\lambda_{far}}{\mathcal{M}\mathcal{L}} \sum_{j=1}^{\mathcal{M}} \sum_{i=1}^{\mathcal{L}} H(f_j \cdot f_i), \quad (4)$$

where H denotes the sigmoid function. We compute the cosine similarity of features for the closest $\mathcal{K} = 2$ and the farthest $\mathcal{L} = 5$ Gaussians. Empirically, we found $\lambda_{near} = 0.05$ and $\lambda_{far} = 0.15$ to yield the best result.

Loss Function. The losses defined in this section are combined in a total loss:

$$\mathcal{L} = \mathcal{L}_{rendering} + \lambda_{clustering}\mathcal{L}_{CC} + \mathcal{L}_{regularization}, \quad (5)$$

where $\mathcal{L}_{rendering}$ is the original rendering loss of 3DGS. Empirically, we set $\lambda_{clustering} = 1 \times 10^{-6}$. See Sect. 4.4 for ablation of these parameters.

4 Experiments

We aim to segment objects within a scene into distinct clusters, to generate novel segmentation masks from any viewpoint of the scene. We therefore compare our algorithm against recent work for scene understanding, which code has already been published. Specifically, we compare our approach against three relevant competitors: LERF [17], Gaussian Grouping [37], and LangSplat [31]. LERF is an open-vocabulary localization method that embeds a language field within a NeRF by grounding CLIP embeddings extracted at multiple scales over the training images. Given a text query, LERF predicts 3D regions with the semantic content pertinent to the input query. The recent Gaussian Grouping [37] is a technique for classifying 3D Gaussians into predefined instances, and LangSplat [31] is an approach that results in a collection of 3D Language Gaussians, such as LERF, outputs a relevancy map for a given text. We evaluate the performance using two metrics: the mean intersection over union (mIoU), which measures the overlap of the GT and rendered masks; and the mean boundary intersection over union (mBIoU), which evaluates contour alignment between predicted and ground truth masks. In both cases, we report the average performance over all test views and text prompts.

In this section, we first provide details about the datasets used to evaluate the models (Sect. 4.1), then give some implementation details (Sect. 4.2) and report the segmentation performance of the models (Sect. 4.3). Finally, we discuss the advantages of a spatial-similarity regularization loss (Sect. 4.4).

4.1 Datasets

We evaluate the chosen models on two datasets containing indoor and outdoor scenes: the LERF-Mask dataset [37] and the 3D-OVS dataset [22].

LERF-Mask. The LERF-Mask dataset is composed of three manually annotated scenes from the LERF-Localization dataset [17]. These scenes belong to the “posed long-tailed objects” of LERF-Localization, which are scenes containing multiple objects with low search volume and low competition, arranged on a plane, like a set of objects arranged on a small table (“Figurines”). These scenes

are captured using the Polycam application on an iPhone, utilizing its onboard SLAM to obtain the camera poses.

3D-OVS. We also report quantitative and qualitative results on five scenes of the 3D-OVS dataset [22], which also consists of a set of long-tail objects, such as toys and everyday objects on a “Bed” or on a “Sofa”.

4.2 Implementation Details

The models evaluated in this section are supervised on segmentation masks automatically generated with the ViT-H SAM model, trained on the SA-1B dataset [18]. These masks are used to learn feature vectors in \mathbb{R}^{16} for each Gaussian. To ensure a stable training process, the loss terms of Eq. (5) are applied with different frequencies: the standard 3DGS loss, used to optimize the geometrical and appearance aspects of the scene, is used at every training iteration. The contrastive clustering loss every 50 iterations, and the spatial-similarity regularization every 100 iterations. Moreover, to reduce the size of the problem and make the loss more stable, we evaluate the clustering loss only on clusters composed by more than 100 features. The optimization of a single scene takes approximately 30k iterations on an NVIDIA 4090 GPU, which amounts to approximately 20 min. The trained model then can render a novel segmentation mask in 0.005 seconds; comparing this with the time necessary to run ViT-H SAM on an image (5.1 sec), this highlights the advantage of the proposed method.

Instance Segmentation. After optimization, the model can be used for *Object Selection*, as exemplified in Fig. 1; given one calibrated image, we want to find the segmentation mask associated with a given selected pixel. Given a 2D pixel location in the image, we obtain a discriminative feature, *i.e.* the rendered feature vector at that pixel’s location. We then generate a 2D similarity map S_C by rendering segmentation features for all pixels of the image, and evaluating their cosine similarity to the discriminative vector. Each pixel of the view $(u, v) \in I$ is then categorized as part of the object of interest or not. Pixels with cosine similarity greater than a fixed threshold t (empirically chosen as $t = 0.7$) are classified as part of the object; otherwise, they are not. The segmentation mask M_{OBJ} is defined as:

$$M_{OBJ}(u, v) = \begin{cases} 1, & \text{if } S_C(u, v) \geq t \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

We note that this process can be applied in parallel to multiple objects, by extracting a set of discriminative features at different locations. An analogous approach also allows the 3D segmentation of the scene, by selecting one or more Gaussians and extracting, for each, all Gaussians with a high similarity score.

Semantic Segmentation. The proposed model renders novel feature maps by projecting and blending the content of the 3D feature field on an image plane. To compare these against the ground truth mask, we follow this procedure: *i)* we select a text prompt related to the content of the scene; *ii)* we feed into Grounding DINO [23] an (Image, Text) pair which provides a bounding box that we use to generate a segmentation mask by using it as a prompt to SAM; *iii)* we sample the rendered feature map associated to a pixel within the segment, and *iv)* use it as a discriminative feature, generating the object’s segmentation in an arbitrary view by selecting all pixels whose rendered-feature vector is falls within a predefined threshold from the discriminative feature.

4.3 Evaluation on Features

First, we compare the performance of our Contrastive Gaussian Clustering against its competitors. We report the average performance on each scene, but a complete breakdown of the performance on each object is available in the Supp.Mat.

Table 1. Comparison of semantic segmentation on LERF-Mask dataset. We report the mIoU and mBIoU (higher is better). LERF-Mask dataset contains accurate segmentation masks that we use to evaluate our segmentation performance.

Method	Figurines		Ramen		Teatime		Average	
	mIoU	mBIoU	mIoU	mBIoU	mIoU	mBIoU	mIoU	mBIoU
LERF [17]	33.5	30.6	28.3	14.7	49.7	42.6	37.2	29.3
Gaussian Grouping [37]	69.7	67.9	77.0	68.7	<u>71.7</u>	<u>66.1</u>	<u>72.8</u>	<u>67.6</u>
LangSplat [31]	44.3	41.9	34.8	28.7	54.3	48.8	44.5	39.8
Ours	91.6	88.8	<u>68.7</u>	<u>63.1</u>	80.5	78.9	80.3	76.9

Table 2. Comparison of semantic segmentation on 3D-OVS dataset, on scenes with sparse long-tail objects and simple background. We report the mIoU (higher is better).

Method	Bed	Bench	Room	Sofa	Lawn	Average
	mIoU	mIoU	mIoU	mIoU	mIoU	mIoU
LERF [17]	73.5	53.2	46.6	27.0	73.7	54.8
Gaussian Grouping [37]	97.3	73.7	<u>79.0</u>	68.1	96.5	<u>82.9</u>
LangSplat [31]	34.3	<u>84.8</u>	56.3	<u>67.7</u>	<u>95.8</u>	67.8
Ours	<u>95.2</u>	96.1	86.8	67.5	91.8	87.5

As shown in Table 1, our method significantly outperforms the other approaches on both metrics, providing on average a +43% accuracy than LERF, +36% accuracy than LangSplat, and +8% accuracy than Gaussian Grouping on average. Regarding the boundary quality of the masks, we outperform on average the competitors by 48%, 37%, and 9%. Though Gaussian Grouping achieves better performance on *Ramen*, we suggest looking at Fig. 3, in which we show how our method produces better qualitative results, with more accurate segmentations.

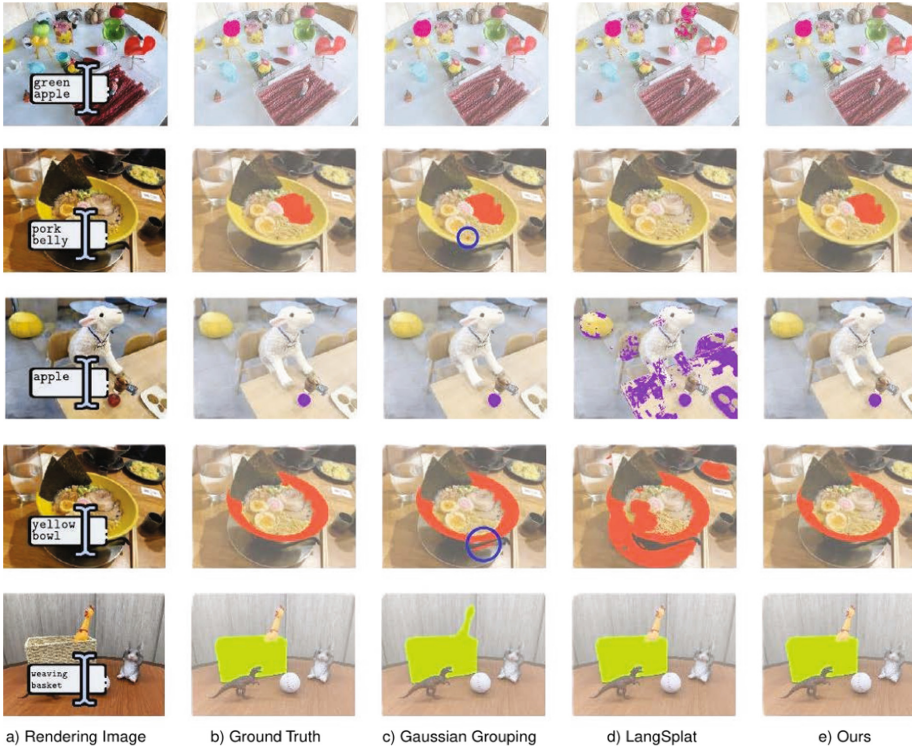


Fig. 3. Qualitative comparison of test views for scenes on LERF-Mask dataset. Our method is able to generate accurate instance segmentation masks for any object on in-the-wild scenes. We replicate and exceed the results in *green apple*, *pork belly*, *apple*. LangSplat exhibits noisy segmentation mask for *old-camera* and coarse segmentation for *sheep*. Gaussian Grouping misclassified some pixels outside *yellow bowl* or classify two objects in the same category in *waving basket*. (Color figure online)

When we test the models on the 3D-OVS, the performance is comparable with the previous experiments, as shown in Table 2. Here, our method outperforms the competitors only on two out of five scenes. This is due to two types of error in the training data masks: type I) incorrect object localization by Grounding

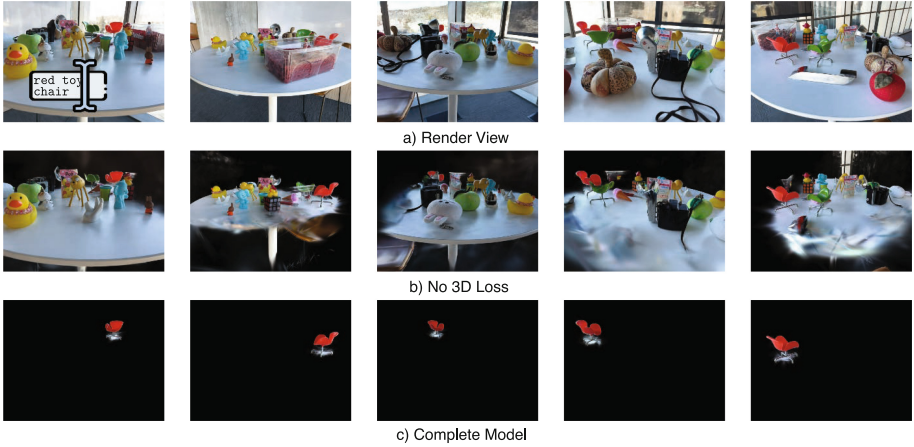


Fig. 4. In this experiment, we extract the Gaussians that belong to the *red toy chair*. We first compute a discriminative feature, following the same procedure as described in Sect. 4.2. Then, we filter the 3D Gaussians by computing its similarity score. The final result is the 3D segmentation of the *red toy chair*. Observe that without our spatial-similarity regularization loss, the 3D segmentation is affected by a high number of outliers. Though these outliers can be easily removed by modifying the similarity threshold, we point-out that the outliers for a fixing similarity threshold is minimum when we use our spatial-similarity regularization loss. (Color figure online)

DINO, and type *II*) incorrect object segmentation by SAM. For example, the average accuracy on *Sofa* is low because of two outlier objects: object *Pikachu*, with a completely incorrect segmentation (Type *I* error), and *grey sofa*, that in most training views is detected as two objects (Type *II* error). However, on average our model achieves the best performances; we outperform LERF on all scenes; and when we perform worse than Gaussian Grouping or LangSplat the performance gap on the mIoU is small: 2.1% on *Bed*, 0.6 on *Sofa*, and 4.7 on *Lawn*.

Of the competitor models, Gaussian Grouping is the one that achieves the closest performance to us. The main limitation of this method is that, while it also enforces multi-view consistency, it does so through preprocessing, requiring that the 2D segmentation masks are made consistent. However, errors in this process propagate to the model, resulting in worse performance. In contrast, our model is not affected by this problem, as it autonomously learns to enforce consistency across the various views. The limited performance of LangSplat is instead due to its embedding in the image semantic features, embedded as 3-dimensional vector, without having a mechanism to ensure no two segments have similar features; this results in noisy segmentation masks and misdetections. This does not happen in our method, since the contrastive clustering loss ensures features from different segments are far in feature space.

Finally, Fig. 3 provides a qualitative comparison of the methods. We can see that the resulting segmentation masks are compatible with the numerical results, showing how our method produces qualitatively better instance segmentations than our competitors. Additional results showing the qualitative performance on 3D segmentation are available in the Supplementary Material.

4.4 Ablation Studies

In the previous experiments, we have claimed that the advantage of our method and, to a lesser extent, of Gaussian Grouping on the other methods is due to the implicitly learned multi-view consistency; which, in our case, is enforced through the loss of Eq. (5). To validate this assumption, we run an ablation test comparing the performance of our model with and without spatial-similarity regularization. The results, reported in Table 3, show that in most scenes the spatial-similarity loss results in a significant performance improvement, on average of 78.8% against 80.3%. This is also supported by the qualitative results on 3D segmentation reported in Fig. 4.

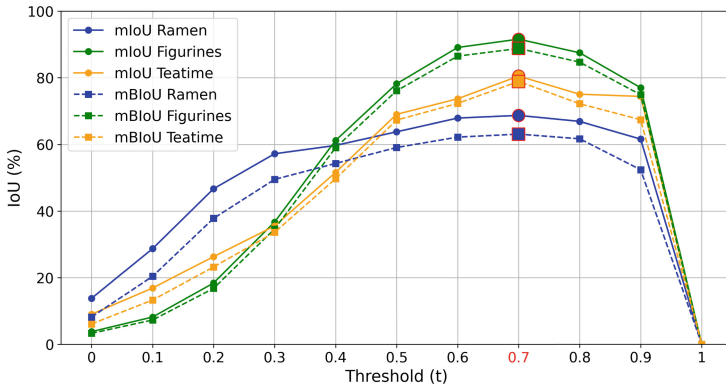


Fig. 5. Segmentation accuracy as a function of threshold t on the LERF-Mask dataset.

Table 3. An ablation study of our model. In this experiment, we explore the impact of the spatial-similarity regularization loss on the segmentation quality. Metrics are averaged over all the test views.

Method	Figurines		Ramen		Teatime		Average	
	mIoU	mBlOU	mIoU	mBlOU	mIoU	mBlOU	mIoU	mBlOU
No 3D Loss	<u>91.7</u>	88.4	<u>67.6</u>	<u>62.4</u>	77.2	73.9	78.8	74.9
No Similarity Loss	92.2	89.2	67.1	62.2	75.3	73.9	78.2	75.1
No Dissimilarity Loss	91.4	88.6	67.1	61.9	<u>80.0</u>	<u>78.6</u>	<u>79.5</u>	<u>76.4</u>
Complete Model	91.6	<u>88.8</u>	68.7	63.1	80.6	78.9	80.3	76.9

Finally, we validate the choice of hyperparameters by studying their effect on segmentation accuracy. Figure 5 shows how setting the instance segmentation

threshold to $t = 0.7$ maximizes performance on all scenes. In Table 4 we instead report the average performance when perturbing each of the hyperparameters of the loss function defined in Eq. (5).

Table 4. Segmentation accuracy as a function of the hyperparameters of Eq. 5.

Hyperparameter	Value	Metrics		Hyperparameter	Value	Metrics	
		mIoU	mBIOU			mIoU	mBIOU
ϵ	10	<u>78.6</u>	<u>75.9</u>	λ_{far}	0.015	<u>77.6</u>	<u>74.6</u>
	100	80.3	76.9		0.15	80.3	76.9
	1000	76.7	72.8		1.5	73.1	70.0
$\lambda_{clustering}$	1×10^{-7}	73.5	70.7	15.0	71.4	68.7	
	1×10^{-6}	80.3	76.9	\mathcal{K}	2	80.3	76.9
	1×10^{-5}	77.8	75.1		5	<u>78.6</u>	<u>75.9</u>
	1×10^{-4}	<u>78.1</u>	<u>75.4</u>		15	77.4	74.3
λ_{near}	0.005	<u>77.6</u>	<u>74.7</u>	\mathcal{L}	2	<u>77.6</u>	<u>74.5</u>
	0.05	80.3	76.9		5	80.3	76.9
	0.5	74.4	71.6		15	73.6	71.2
	5.0	77.0	73.2				

5 Conclusions

In this paper, we introduce Contrastive Gaussian Clustering, a novel approach for 3D scene segmentation. We have shown how, by implicitly enforcing a contrastive clustering loss, we are able to learn consistent segmentation features from an inconsistent set of 2D segmentation masks. This means that the proposed model can learn from automatically generated segmentation masks, with little to none preprocessing required. Moreover, the use of a spatial-similarity regularization ensures that the features learned for Gaussians corresponding to different 3D clusters are distinct enough to provide accurate 3D segmentation. The combination of such two losses results in an efficient and accurate model that outperforms current approaches based both on NERF and 3DGS.

Limitations. Although the results reported in the paper are very promising, including additional information involves some trade-off. Foremost, the use of the two additional losses involves a computational overhead with respect to standard 3DGS, requiring on average 100% longer time to train. We can, however, reduce this by only applying the losses every 50/100 iterations, respectively. Moreover, the additional information stored in the Gaussians requires larger memory capacity; future works will consider more efficient ways of including the identity information into the scene representation. Other limitations are inherited from SAM and Grounding DINO. For example, to select all Gaussians matched to a given semantic label, we rely on Grounding DINO to select that object’ location in a reference image. However, if this location is wrong, it will

not be possible to recover the correct mask. The model’s performance is also limited by the accuracy of the 2D segmentations used in training. We observe that, if multiple views contain incorrect masks, this can result into multiple instances being clustered together.

Future Works. We will expand the proposed approach, integrating it with LLM for language interaction, and extending the feature field to also include hierarchical segmentations. Future work will explore more advanced ways of contrastive clustering. Concerning our multi-view contrastive loss, in future work we could explore more intelligent ways to contrast all the feature-objects.

Acknowledgments. This project has received funding from the European Union’s Horizon research and innovation programme under grant agreement No 101079116 and No 101079995.

References

1. Barron, J.T., et al.: Mip-NeRF 360: unbounded anti-aliased neural radiance fields. In: CVPR (2022)
2. Behley, J., et al.: SemanticKITTI: a dataset for semantic scene understanding of lidar sequences. ICCV (2019)
3. Bhalgat, Y., Laina, I., Henriques, J.F., Zisserman, A., Vedaldi, A.: Contrastive lift: 3D object instance segmentation by slow-fast contrastive fusion. In: NeurIPS (2023)
4. Caron, M., et al.: Emerging properties in self-supervised vision transformers. In: ICCV (2021)
5. Cen, J., et al.: Segment any 3D Gaussians. arXiv preprint [arXiv:2312.00860](https://arxiv.org/abs/2312.00860) (2023)
6. Chang, A., et al.: Matterport3D: learning from RGB-D data in indoor environments. 3DV (2017)
7. Chen, D.Z., Chang, A.X., Nießner, M.: Scanrefer: 3D object localization in RGB-D scans using natural language. In: ECCV (2020)
8. Chen, G., Wang, W.: A Survey on 3D Gaussian Splatting (2024)
9. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In: CVPR (2017)
10. Fang, J., Wang, J., Zhang, X., Xie, L., Tian, Q.: GaussianEditor: editing 3D Gaussians delicately with text instructions. arXiv preprint [arXiv:2311.16037](https://arxiv.org/abs/2311.16037) (2023)
11. Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V., Garcia-Rodriguez, J.: A review on deep learning techniques applied to semantic segmentation (2017)
12. Genova, K., et al.: Learning 3D semantic segmentation with only 2D image supervision. 3DV (2021)
13. Hou, J., Dai, A., Nießner, M.: 3D-SIS: 3D semantic instance segmentation of RGB-D scans. In: CVPR (2019)
14. Hu, Q., et al.: Randla-net: efficient semantic segmentation of large-scale point clouds. In: CVPR (2020)
15. 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: 3DV (2016)
16. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. (2023)

17. Kerr, J., Kim, C.M., Goldberg, K., Kanazawa, A., Tancik, M.: LERF: language embedded radiance fields. In: ICCV (2023)
18. Kirillov, A., et al.: Segment anything. [arXiv:2304.02643](https://arxiv.org/abs/2304.02643) (2023)
19. Kundu, A., et al.: Panoptic neural fields: a semantic object-aware neural scene representation. In: CVPR (2022)
20. Li, B., Weinberger, K.Q., Belongie, S., Koltun, V., Ranftl, R.: Language-driven semantic segmentation. In: ICLR (2022)
21. Liao, Y., Xie, J., Geiger, A.: KITTI-360: a novel dataset and benchmarks for urban scene understanding in 2D and 3D. TPAMI (2023)
22. Liu, K., et al.: Weakly supervised 3D open-vocabulary segmentation. In: NeurIPS (2023)
23. Liu, S., et al.: Grounding DINO: marrying DINO with grounded pre-training for open-set object detection. arXiv preprint [arXiv:2303.05499](https://arxiv.org/abs/2303.05499) (2023)
24. Liu, Y., Fan, Q., Zhang, S., Dong, H., Funkhouser, T.A., Yi, L.: Contrastive multimodal fusion with tupleinfonce. ICCV (2021)
25. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
26. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. (2022)
27. Naseer, M., Khan, S., Porikli, F.: Indoor scene understanding in 2.5/3D for autonomous agents: a survey. IEEE Access (2019)
28. Peng, S., Genova, K., Jiang, C.M., Tagliasacchi, A., Pollefeys, M., Funkhouser, T.: OpenScene: 3D scene understanding with open vocabularies (2023)
29. Qi, C.R., et al.: PointNet++: deep hierarchical feature learning on point sets in a metric space. In: NeurIPS (2017)
30. Qi, C., Su, H., Mo, K., Guibas, L.: Pointnet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 652–660 (2017)
31. Qin, M., Li, W., Zhou, J., Wang, H., Pfister, H.: LangSplat: 3D language Gaussian splatting (2023)
32. Radford, A., et al.: Learning transferable visual models from natural language supervision. In: Proceedings of the 38th International Conference on Machine Learning, pp. 8748–8763. PMLR (2021)
33. Sautier, C., Puy, G., Gidaris, S., Boulch, A., Bursuc, A., Marlet, R.: Image-to-lidar self-supervised distillation for autonomous driving data. In: CVPR (2022)
34. Kobayashi, S., Matsumoto, E., Sitzmann, V.: Decomposing NeRF for editing via feature field distillation. In: NeurIPS (2022)
35. Wu, G., et al.: 4D Gaussian splatting for real-time dynamic scene rendering. arXiv preprint [arXiv:2310.08528](https://arxiv.org/abs/2310.08528) (2023)
36. Wu, Z., et al.: 3D ShapeNets: a deep representation for volumetric shapes. In: CVPR (2015)
37. Ye, M., Danelljan, M., Yu, F., Ke, L.: Gaussian grouping: segment and edit anything in 3D scenes. arXiv preprint [arXiv:2312.00732](https://arxiv.org/abs/2312.00732) (2023)
38. Ying, H., et al.: Omnise3D: Omniversal 3D segmentation via hierarchical contrastive learning (2023)
39. Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: radiance fields without neural networks (2021)
40. Zhi, S., Laidlow, T., Leutenegger, S., Davison, A.J.: In-place scene labelling and understanding with implicit scene representation. In: ICCV (2021)

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.





AYANet: A Gabor Wavelet-Based and CNN-Based Double Encoder for Building Change Detection in Remote Sensing

Priscilla Indira Osa^{1,2} , Josiane Zerubia² , and Zoltan Kato^{3,4} 

¹ DITEN Department, University of Genoa, Genoa, Italy

² Inria, Université Côte d'Azur, Nice, France

priscilla.indira.osa@edu.unige.it

³ Institute of Informatics, University of Szeged, Szeged, Hungary

kato@inf.u-szeged.hu

⁴ J. Selye University, Komarno, Slovakia

Abstract. The main challenge presents in bitemporal building change detection (BCD) in remote sensing (RS) is to detect the relevant changes that are related to the buildings, while ignoring changes induced by other types of land cover as well as varied environmental condition during the sensing process. In this paper, we propose a new BCD model with a double encoder architecture. The Gabor wavelet-based encoder which aims to highlight the characteristic of buildings on RS imagery i.e., the comparatively more regular and repetitive texture than other objects on RS images. This Gabor Encoder is used in addition to the convolutional-neural-network-based encoder that extracts other meaningful and high-level information from the images. Moreover, we also propose Feature Conjunction Module to efficiently combine the extracted features by characterizing possible types of changes. Comparative results with State-of-the-art models on 3 different BCD datasets (LEVIR-CD, S2Looking, and WHU-CD) confirm that the proposed model outperforms current BCD methods in producing a highly accurate change map of buildings. Our code is available on <https://github.com/Ayana-Inria/AYANet>.

Keywords: Gabor wavelet · Convolutional Neural Network · Building Change Detection · Remote Sensing

1 Introduction

Change Detection aims to identify changes occurred in a scene between two different times, based on a pair of (geometrically) registered images acquired at pre

The first author performed the work while at Inria, Université Côte d'Azur, France. University of Genoa and Université Côte d'Azur are part of the Ulysseus Alliance (European University). <https://ulyseus.eu/>.

and post-event. Some examples of the event that can cause the changes include urban expansion, deforestation, or natural disaster. The challenge is to recognize changes attributed to the event, while ignoring any other visual changes that are unrelated to the event itself, which are generally due to lighting conditions, shadows, seasonal variations, or changes in other environmental conditions. An important special case of change detection is *Building Change Detection* (BCD), where the goal is to highlight changes only in buildings and ignore the irrelevant changes of other objects (*e.g.* vegetation) [22]. In remote sensing imagery, built-in regions typically have a distinctive repetitive visual pattern compared to other natural regions. Thus, such characteristics are important in identifying built-in areas as well as any changes related to buildings (either buildings that are demolished or newly constructed). The typical BCD task creates a change map that highlights appearance and disappearance of buildings, which can be used as the starting point of a broad range of applications, such as urban growth analysis [9], and disaster assessment and recovery [20].

Traditional change detection methods can be divided into pixel-based and object-based methods. While pixel-based methods rely on pixel-wise spectral value changes between bi-temporal images, object-based methods can incorporate both spectral and spatial (*e.g.* shape, texture) contextual information of images. The former approach is challenged by the limited spatial contextual information provided by a small neighborhood of pixels, while the latter one is subject to object segmentation errors and lacks the capability to include both local and global features which are crucial as local features preserve spatial details and global features provide a bigger context information to accurately recognize the semantic information of pixels.

Deep Convolutional Neural Networks (CNN) have demonstrated promising performance in addressing the complexities of the BCD task [1, 4, 13, 18, 26]. CNN is able to extract image features via spatial convolutions and hierarchical feature representations, which successfully combines local features by gradually increasing the effective receptive field of subsequent layers as it goes deeper in the network, creating a pyramid-like stack of features at multiple resolution. Recently, Transformer networks are becoming popular in BCD because of their efficiency in capturing the global context of the features. It can be incorporated in combination with CNN [3, 15], or it can also be used without feature extraction by CNNs [2]. Theoretically, both CNN and Transformer can learn texture features from the training image data [17, 19], assuming sufficiently many training data are available. However this is not the case in remote sensing imagery. While general purpose large datasets exist to train such networks, *e.g.* ImageNet [8] which contains around 14 million images, and JFT-3B [27] with approximately 3 billion images, open BCD datasets generally contain fewer images by several order of magnitude (less than hundreds of thousands). This is a serious constraint when more and more complex models are appearing with several million parameters to learn.

Models based on CNN, Transformer, or both, incorporate typical strategies such as metric-based learning [4], as well as integrating attention mechanisms [1,

4, 10, 18]. Indeed, attention-based approaches put weights on relevant features *e.g.* temporal attention which emphasizes the relation between the features of the bi-temporal images that accentuate the change [4]. However, considering the typical size of BCD datasets, it is by far not evident that such complex networks can learn features effectively, especially Transformer which may fail to learn some specific features if the training data are not provided sufficiently [19]. On the other hand, other approaches are using fewer parameters to learn by reducing the complexity of the network [3, 7, 10, 15]. While all of the State-of-the-Art methods mentioned above, including attention-based, Transformer-based or CNN-based ones, perform well (see Sect. 3.3), none of them explicitly perform feature extractions that are characteristic to the particular texture properties of building in spite of its importance in differentiating buildings from other objects in the BCD task.

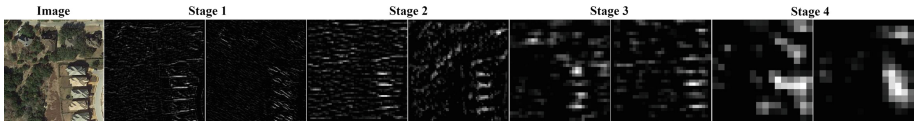


Fig. 1. Some examples of features extracted at different stages of the Gabor Encoder. The deeper the stage goes, the lower the resolution is. Notice that regions with buildings are clearly highlighted at each resolution, while other regions (in spite of being textured but without regularity) are suppressed in the feature maps.

To address this issue, we propose AYANet which adopts a double-encoder feature extraction backbone that provides rich *texture* features in a Siamese network to extract multi-scale features from bi-temporal image pairs. At each resolution, feature differences are extracted and forwarded to a final decoder, which identifies building changes and provides the final change map. The main contributions of this paper are:

1. We integrate local feature extraction from a CNN-based encoder which is based on EfficientNet-B7 [23], and explore the advantages of a dedicated multi-scale texture feature encoder based on Gabor wavelets [11], in the form of a so called *double encoder* where CNN-extracted hierarchical features are augmented by features directly representing repetitive visual patterns at different scale and orientation. One can also interpret it as a kind of attention to highly regular textured regions. Figure 1 illustrates some multi-scale features extracted by the proposed Gabor Encoder. We can observe that the extracted features highlight the textures of buildings that are located on the right side of the image. While a CNN can already extract general texture features from the input images, the intuition we have in mind when designing the Gabor Encoder is to ensure the encoder to extract the textures belong to the buildings by imposing Gabor filters when the network learns to update the convolutional filters which are integrated together as the building block of the encoder (more detail in Sect. 2.1).

- Features from corresponding scale in the Siamese network are processed by a Feature Conjunction Module (FCM), which will characterize their dissimilarity for the decoder.

The quantitative and qualitative experimental results on standard datasets demonstrate the superiority of our method.

2 AYANet

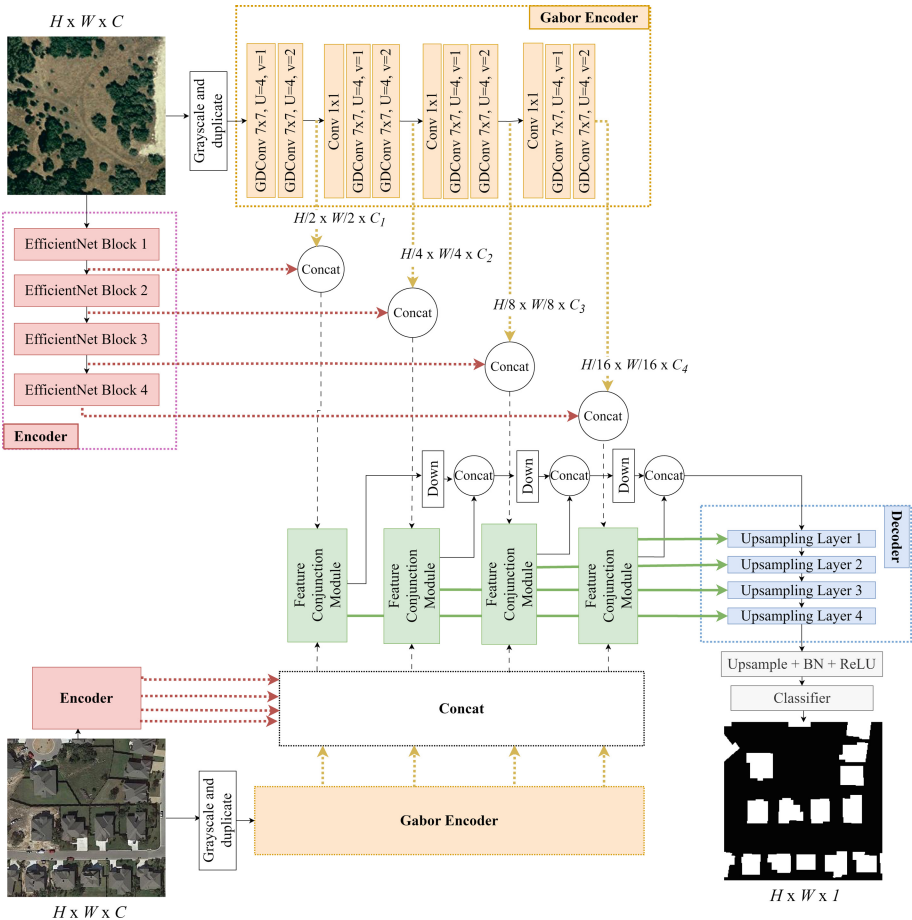


Fig. 2. The architecture of AYANet. The design follows the style of a Siamese network i.e., the same Encoder and Gabor Encoder are used to process the two input images.

The proposed model, shown in Fig. 2, is a Siamese network with three main components:

1. double encoder consisting of our *Gabor Encoder* and the EfficientNet-based general *Encoder*
2. *Feature Conjunction Modules* at 4 resolutions
3. the *Decoder* and *Classifier* which produces the final change map

A pair of pre-change and post-change images with input size $H \times W \times C$ (H, W, C refer to height, width, channels respectively), goes directly to the Encoder. The Encoder produces multi-scale features from each block with a size of $\frac{H}{2^i} \times \frac{W}{2^i} \times C_i$, where $i = \{1, 2, 3, 4\}$, and $C_{i+1} > C_i$. The same pair of input images is converted to grayscale and is duplicated to $H \times W \times C_1$ before being fed to the Gabor Encoder in order to accommodate the depthwise mechanism used in that block, which will be explained in detailed in Subsect. 2.1. The Gabor Encoder extracts features at different scales at the same resolution as the features extracted by the Encoder. Features are then concatenated and being passed to the Feature Conjunction Module where pre-change and post-change features are combined such that feature changes are highlighted. These conjugated features are subsequently passed to the Decoder. The Decoder of AYANet utilizes the decoder part of [5], which comprises several upsampling layers. The operation includes a simple bilinear upsample followed by the sum of upsampled features, and the features coming directly from the FCM module at every stage. The binary change map is produced by classifying the features upsampled by transposed convolution layers.

2.1 Double Encoder

Feature extraction by the double encoder comprises two components. One CNN-based Encoder, which consists of the first 4 mobile inverted bottleneck blocks (MBConv) of EfficientNet-B7 [23]. The depthwise separable convolution implementation in the building block of EfficientNet allows deep feature extraction with less computational cost compared to architectures using regular convolution blocks. Moreover, the squeeze and excitation (SE) block [12] in MBConv will act as the channel attention mechanism in the Encoder which models the interdependencies among channels of the features. The other one is our Gabor Encoder which focuses on the repetitive visual patterns of the buildings.

Gabor Encoder. The main element of Gabor Encoder is inspired by Gabor Orientation Filters (GoFs) proposed in [17]. A GoF consists of a group of filters in which each of the filter is a learnable convolutional filter modulated by a Gabor filter [17]. Gabor filters [11] are biologically motivated as mammals' vision system uses similar multiscale filters to extract texture information from retinal images. Gabor filters are represented by the following equation [16, 17, 25]:

$$G(u, v) = \frac{\|\mathbf{k}_{u,v}\|^2}{\sigma^2} e^{-(\|\mathbf{k}_{u,v}\|^2 \|z\|^2 / 2\sigma^2)} [e^{i\mathbf{k}_{u,v} \cdot z} - e^{-\sigma^2/2}], \quad (1)$$

where $\mathbf{z} = (x, y)$, $\mathbf{k}_{u,v} = \begin{pmatrix} k_{jx} \\ k_{jy} \end{pmatrix} = \begin{pmatrix} k_v \cos k_u \\ k_v \sin k_u \end{pmatrix}$, frequency $k_v = (\pi/2)/\sqrt{2}^{(v-1)}$, orientation $k_u = u \frac{\pi}{V}$, and $\sigma = 2\pi$. The scale parameter $v = 1, \dots, V$ controls the

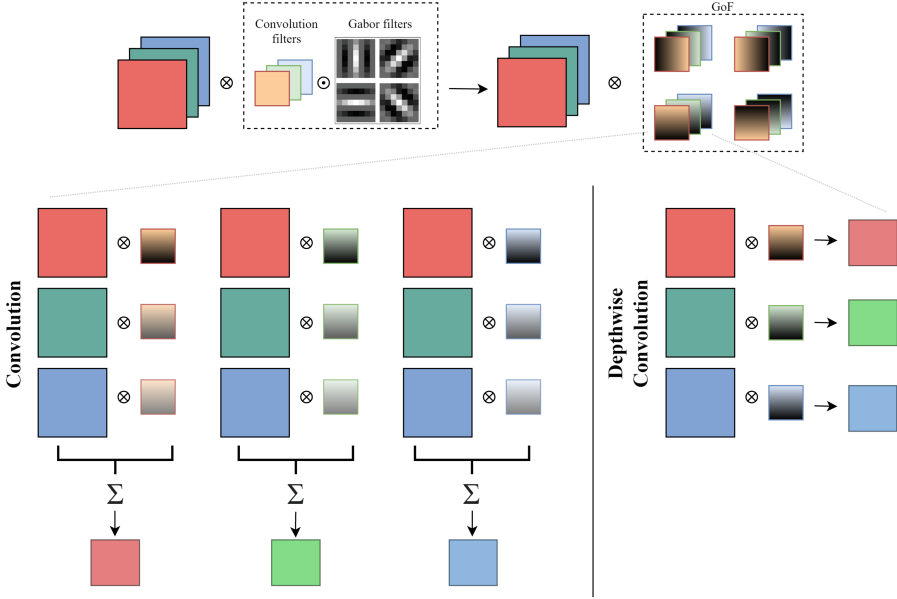


Fig. 3. The upper part of the image shows how GoF is obtained and the lower part indicates the difference on how the filters work between standard convolution (left) and depthwise convolution (right).

frequency of the filter in inverse proportion while the parameter $u = 0, \dots, U - 1$ determines the orientation of the filter.

Each filter in a GoF is a product of element-wise multiplication of a convolutional filter C_i of size $N \times K \times K$ with a Gabor filter $G(u, v)$ with size $K \times K$, orientation u , and scale v

$$C_{i,u}^v = C_i \odot G(u, v), \tag{2}$$

Thus, a GoF [17] comprises a group of filters with a scale v and a set of orientations U

$$\hat{C}_i^v = (C_{i,0}^v, \dots, C_{i,U-1}^v), \tag{3}$$

The upper part of Fig. 3 illustrates the process to obtain a GoF. We intuitively interpret the integration of Gabor filters in the convolutional block, in some way, guides the parameter learning in the Gabor Encoder to be imposed by Gabor filters we set in the GoFs because the backpropagation process will take into account the Gabor filters in each block [17]. Additionally, we modified the original GoF by replacing standard convolution to depthwise convolution [6], which changes the operation (assume stride = 1 with padding) from

$$\hat{F}_{k,l,n} = \sum_{i,j,m} K_{i,j,m,n} F_{k+i-1,l+j-1,m}, \tag{4}$$

to

$$\hat{F}_{k,l,m} = \sum_{i,j} K_{i,j,m} F_{k+i-1,l+j-1,m} \quad (5)$$

where K is the filter, F is the input image or feature, and \hat{F} is the output feature. (i, j) denotes the position of the cell indexed based on the kernel size, (k, l) defines the position of the cell indexed based on the output feature size, m increments until the number of input channel, and n is looped until the number of output channel. The lower part of Fig. 3 shows the difference between convolution operation on the left and depthwise convolution on the right. Depthwise convolution applies one kernel to each input channel, which provides the following benefits: 1) less computational cost and 2) filtering the features spatially without the mixing of channel-wise information. In order to implement this, we need to make sure that the number of input channels is the same as the number of filters or the output channels, which is the reason why we need to duplicate the input image to the number of filters of the first block in the Gabor Encoder.

Referring back to Fig. 2, The block of operations between the input image or input features and GoF with the depthwise convolution is called GDConv. Two blocks of GDConv with kernel size of 7×7 are responsible to produce the Gabor Encoder’s output features at a particular resolution. These output features from each stage are then to be concatenated with the output features from the EfficientNet encoder at the same resolution. The orientation of GDConv was set to $U = 4$ to represent the horizontal, vertical, and diagonal orientations. The scale parameters were $v = 1$ and $v = 2$ for the first and second GDConv block respectively. A depthwise convolution with stride 2 is used in the second block to bring down the spatial resolution to half of the input size. In order to adjust the channel size, we implement Convolution 1×1 before every first block of each stage except for the first stage where the channel adjustment is handled by duplicating the image channel. Every second block of each stage is also followed by Batch Normalization and ReLU activation function.

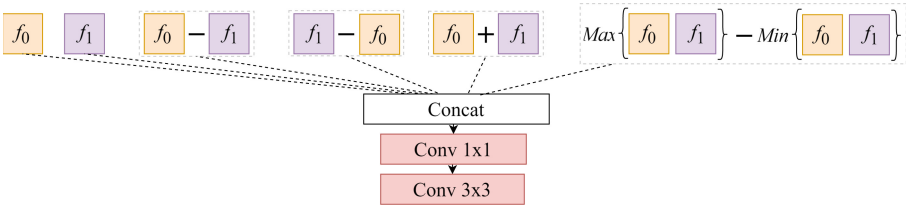


Fig. 4. The structure of Feature Conjunction Module.

2.2 Feature Conjunction Module

The extracted multi-resolution features concatenated from both encoders are processed by the Feature Conjunction Module (FCM). As shown in Fig. 4,

we treat pre-change feature f_0 and post-change feature f_1 with several operations similar to [24], in order to explicitly represent the behavior of bi-temporal changes. Referenced from [24], $f_0 - f_1$ and $f_1 - f_0$ define the appearance and disappearance of the object, while $Max(f_0, f_1) - Min(f_0, f_1)$ intends to capture exchanging objects. We additionally add $f_0 + f_1$ to highlight the changed objects from the unchanged ones. All of the products of these operations are concatenated together with the original features f_0 and f_1 , then they undergo a 1×1 convolution which will learn the important channels related to the changes and reduce the resolution from $\frac{H}{2^i} \times \frac{W}{2^i} \times 6C_i$ to $\frac{H}{2^i} \times \frac{W}{2^i} \times C_i$. A 3×3 convolution followed by Batch Normalization and ReLU activation are added as the last stage to further learn the relevant features.

3 Experiments

The performance of AYANet has been evaluated on 3 standard RS building change detection datasets. Comparison with State-of-the-Art (SOTA) methods is done both quantitatively using standard metrics, and qualitatively by visualizing the change maps. Some of the SOTA methods have been trained and tested in-house to ensure a fair comparison, while for other methods we report the measurements on the standard test split of the datasets published in papers.

3.1 Datasets

LEVIR-CD [4] consists of very high-resolution (VHR) RGB imagery highlighting the change in the development as well as the decline of buildings in Texas, USA. The dataset has 31333 change instances of various types of buildings such as large warehouses, tall apartments, villa residences, and small garages. For the experiment, we cropped 637 pairs of 0.5m resolution images with a size of 1024×1024 pixels to 256×256 patches without overlap. Following the default split of the dataset, the total pairs used for training/validation/test is 7120/1024/2048.

S2Looking [21] has 5000 bitemporal VHR side-viewing satellite imagery obtained at several off-nadir angles. The images are captured from various satellites such as GaoFen, SuperView, and BeiJing-2 with a size of 1024×1024 pixels and spatial resolutions ranging from 0.5m to 0.8m. The S2Looking dataset contains scenes of rural areas from around the world which adds the complexity of features of the dataset. The default split of train/validation/test consists of 3500/500/1000 pairs of images. For the experiment, the images were cropped into 256×256 patches which makes the final split adds up to 56000/8000/16000.

WHU-CD [14] records the building changes in Christchurch, New Zealand between 2012 and 2016. This dataset contains a pair of RGB aerial images with 0.2m spatial resolution. The training split has a size of 21243×15354 pixels and the test split is 11256×15354 pixels. Like the other two datasets, the images were cropped to 256×256 , and we randomly split the images to 6096/762/762 for train/validation/test.

3.2 Implementation Details

The implementation of the proposed model was done using PyTorch and we run experiments on two GPUs: NVIDIA Quadro GV100 and NDVIA GeForce RTX4090. Input images were augmented geometrically (random flipping, random cropping) or photometrically (Gaussian blur). Weights of the model were randomly initialized. We trained the model using cross-entropy loss and AdamW optimizer (weight decay 0.01 and beta values (0.9, 0.999)). The model started the training with learning rate from 0.0001 linearly decaying to 0. We set the batch size to 8 and stopped the training at 300 epochs.

We utilized Precision, Recall, F1-score, and Intersection over Union (IoU) for the quantitative evaluation of our model.

Table 1. Quantitative results of AYANet and State-of-The-Art models on the LEVIR-CD dataset. The best result is highlighted in bold. Results of all SOTA models are as reported in the original papers.

Model	Precision	Recall	F1-score	IoU
AFCF3D-Net [26]	91.35%	90.17%	90.76%	83.08%
BIT [3]	89.24%	89.37%	89.31%	80.68%
ChangeFormer [2]	92.05%	88.80%	90.40%	82.48%
DML-Net [10]	92.52%	89.95%	90.71%	82.99%
DUNE-CD [1]	92.27%	88.83%	90.52%	82.68%
FHD [18]	92.61%	89.61%	91.09%	83.63%
GVA-CD [13]	92.63%	87.88%	90.31%	82.51%
MSFCTNet [15]	92.06%	90.00%	91.02%	83.52%
STANet-PAM [4]	83.81%	91.00%	87.26%	77.40%
TINYCD [7]	92.68%	89.47%	91.05%	83.57%
AYANet	92.60%	90.25%	91.41%	84.17%

3.3 Comparison with SOTA

We listed the comparison of performances among our proposed model and several SOTA models on the LEVIR-CD dataset in Table 1. We make use of the default train/validation/test split, which has been used by many papers to report their results as well - which allows us a direct comparison with numerous SOTA methods. Furthermore, we only report results published in the original paper of the methods (which are the optimized results of the authors themselves) to guarantee a fair comparison with our method. The SOTA methods listed in Table 1 represent a broad range of techniques and strategies. AFCF3D-Net [26] treats bitemporal images like a video and uses 3D CNN as its backbone. CNN-based models such as DML-Net [10], DUNE-CD [1], FHD [18], STANet-PAM [4], and

TINYCD [7] incorporate various attention mechanisms including self-, channel-, global-, local-, and cross-attention. GVA-CD [13] focuses on the feature difference method by taking into account the geometric structure of the object. BIT [3], ChangeFormer [2], and MSFCTNet [15] utilizes Transformer either in hybrid style or using it purely without CNN.

It can be observed that AYANet performs better than most of the listed SOTA models and outperforms all in terms of F1-score and IoU. This includes surpassing the models that implement Transformer, for example the CNN-Transformer-hybrid BIT by 2.10% and 3.49%, and the pure Transformer-based ChangeFormer by 1.01% and 1.69%. The proposed model also exceeds the performance of TINYCD by 0.36% and 0.60%. TINYCD also uses EfficientNet as their feature extractor, and a more sophisticated technique to manipulate the features extracted, as opposed to a simpler operation used in our FCM. We intuitively correlate this outcome to the addition of the Gabor Encoder helping extracting relevant features of the buildings such that only a simple feature manipulation is necessary to highlight the change of the buildings. Comparing to the method that explicitly target the pattern of the object on the image *i.e.* GVA-CD which focuses on geometric variation, our proposed model which targets building’s textures, has a 1.10% higher F1-score and a 1.66% higher IoU.

Table 2. Quantitative results of AYANet and State-of-The-Art models on the S2Looking dataset. The best result is highlighted in bold. All SOTA models’ results are reproduced.

Model	Precision	Recall	F1-score	IoU
BIT [3]	73.99%	52.73%	61.58%	44.49%
ChangeFormer [2]	68.04%	57.03%	62.05%	44.98%
STANet-PAM [4]	36.30%	61.84%	45.74%	29.65%
AYANet	69.37%	58.70%	63.59%	46.62%

Table 3. Quantitative results of AYANet and State-of-The-Art models on the WHU-CD dataset. The best result is highlighted in bold. All SOTA models’ results are reproduced.

Model	Precision	Recall	F1-score	IoU
BIT [3]	87.65%	90.91%	89.25%	80.59%
ChangeFormer [2]	94.15%	85.52%	89.63%	81.20%
STANet-PAM [4]	70.65%	93.54%	80.50%	67.37%
AYANet	95.56%	92.89%	94.21%	89.05%

The evaluation on the S2Looking dataset and the WHU-CD dataset were also done. The S2Looking dataset covers a more challenging task where images are

taken from off-nadir angles. Perhaps for this reason, relatively few papers report evaluation results on this difficult dataset and the reported IoU numbers are all below 50%. The WHU-CD dataset does not have a standard train/val/test split such that most of the literature present their results by randomly splitting the set. Unlike on the LEVIR-CD dataset, a fair comparison thus cannot be done based on only the published numbers. Therefore we re-trained and evaluated relevant SOTA methods on the same split of this dataset. We selected three models representing pure CNN (STANet-PAM), hybrid CNN and Transformer (BIT), and pure Transformer models (ChangeFormer). Table 2 and Table 3 show the quantitative results of AYANet and SOTA models on the S2Looking and the WHU-CD datasets respectively. The proposed model shows the highest performance in terms of F1-score and IoU among the evaluated methods on both datasets. There are minimal differences of 1.54% and 1.64% on the S2Looking dataset as well as significant improvement by 4.58% and 7.85% on the WHU-CD dataset, w.r.t. the second-best performer model *i.e.* ChangeFormer.

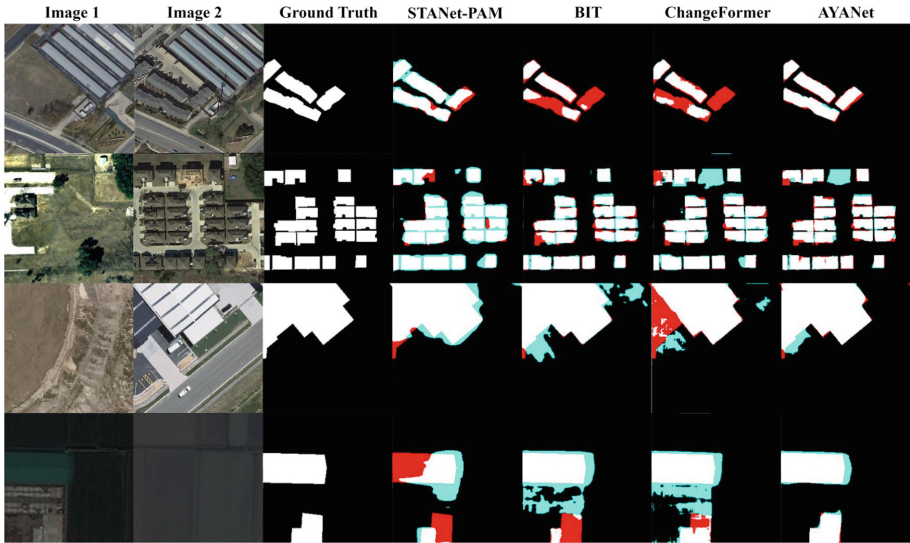


Fig. 5. Qualitative comparison of the change maps predicted by the proposed model and the SOTA models. The first 2 rows are the results on LEVIR-CD, while the third and fourth rows are from the WHU-CD and S2Looking datasets respectively. Color representation: TP (white), FP (light blue), TN (black), FN (red). (Color figure online)

The qualitative comparison is shown in Fig. 5 where it can be seen that AYANet’s change maps have less false positive (light blue area) and false negative (red area) in several cases, such as detecting changes of big building on the third row of the figure, recognizing changes in smaller buildings on the first row, as well as change detection in the environment with poor lighting condition shown in the last row of the figure. Moreover, our model produces more precise masks like

what we can observe in the second row of the figure where the boundary of the buildings located close to each other appears to be clearer.

We did an additional experiment to test our models that were trained on one particular dataset, with another datasets. The goal of this experiment is to show the proxy of generalization ability of the models. Results in Table 4 show the performance of the models trained on LEVIR-CD and tested on the WHU-CD dataset. The proposed model outperforms other models in F1-score and IoU at least by 2.28% and 2.85%. The difference can also be confirmed in Fig. 6 where we can observe that AYANet’s change maps have less false positive and false negative prediction. However, note that even our best performing model reaches only 60%, which is obviously much lower than anything trained on the WHU-CD dataset itself. Other SOTA models also reported the same tendency which may be related to the rather large difference in remote sensing imagery making change detection methods generalization challenging, partially because the pre and post-images are already registered so change detection requires only a pixel-wise analysis of changes thus more global changes are not learned well by these models and this is not even their goal to do so.

Table 4. The results of cross-dataset evaluation. All models are trained on the LEVIR-CD dataset and are tested on the WHU-CD dataset.

Model	Precision	Recall	F1-score	IoU
BIT [3]	58.36%	79.52%	67.32%	50.74%
ChangeFormer [2]	76.87%	70.10%	73.33%	57.89%
STANet-PAM [4]	28.31%	14.27%	18.98%	10.48%
AYANet	77.60%	73.66%	75.58%	60.74%

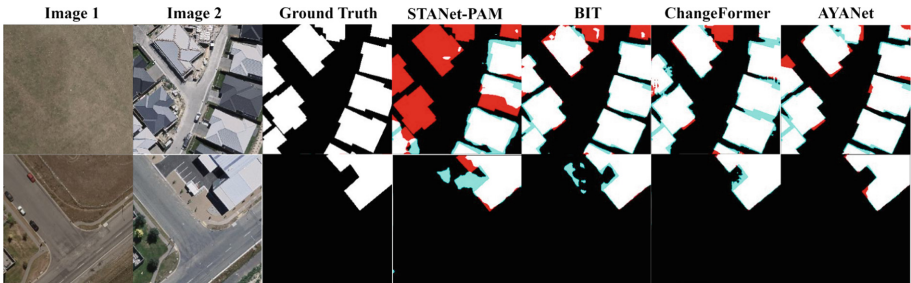


Fig. 6. Qualitative performance of the models on cross-dataset evaluation. Color representation: TP (white), FP (light blue), TN (black), FN (red). (Color figure online)

3.4 Ablation Study

An ablation study was conducted to check how the proposed model behaves according to different settings of encoder. Table 5 shows the qualitative results

of AYANet with the proposed *double encoder*, and the cases where we only use the Gabor Encoder as well as modified EfficientNet we use as the Encoder, exclusively. We find that using only the Gabor Encoder does not give the model a satisfactory performance as it only reaches 88.92% in F1-score, and 80.05% in IoU compared to AYANet which has 91.41% and 84.17% in the same metrics. However, adding the Gabor Encoder to the deep convolutional feature extractor, EfficientNet does increase the result, especially in IoU which implies a better agreement between the area of prediction and the ground truth. Some examples of the predictions shown in Fig. 7 confirm this IoU improvement. It can be seen that AYANet enhances boundary between buildings compared to the cases when we only use one single encoder.

Table 5. The experiments on the encoder of AYANet on the LEVIR-CD dataset.

Encoder	Precision	Recall	F1-score	IoU
Gabor	90.51%	87.38%	88.92%	80.05%
EfficientNet	92.15%	90.35%	91.24%	83.90%
AYANet	92.60%	90.25%	91.41%	84.17%

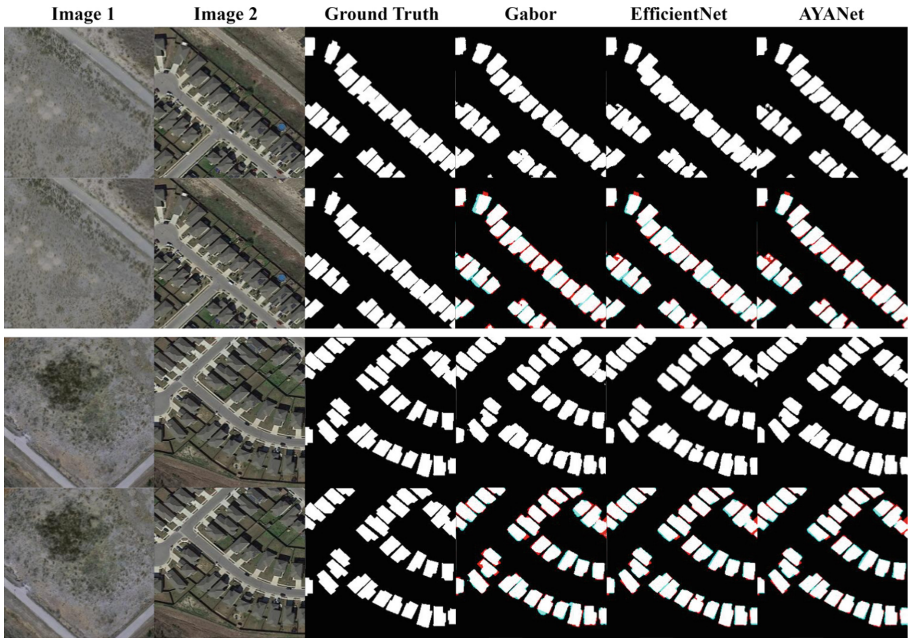


Fig. 7. The visualization of the ablation study on encoder. Visualization is done with and without color representation to make the boundary between buildings more visible. Color representation: TP (white), FP (light blue), TN (black), FN (red). (Color figure online)

4 Conclusion

We introduce AYANet, a remote sensing change detection model using double encoder as the features extractor. The design of the double encoder includes CNN-based encoder and the Gabor Encoder which aims to extract the texture features of buildings. Moreover, Feature Conjunction Module is also proposed to process the extracted features from the double encoder in order to characterize the changes. Based on the comparison with SOTA models and the experimental evaluation, the proposed model demonstrates a good performance on 3 different building change detection datasets that have different characteristics. The ablation study confirms that adding the Gabor Encoder to the CNN-based encoder predicts a more accurate boundary between buildings. Future work will focus on a novel learning strategy that accommodates for domain adaptation, and unsupervised or semi-supervised learning approaches to cater to the problem of limited amount of data.

Acknowledgements. The authors acknowledge the internship funding support by Inria, France (BMI-NF); the grants TKP2021-NVA-09 and K135728 of the National Research, Development and Innovation Fund, Hungary; and the scholarship by French Government for the first author.

References

1. Adil, E., Yang, X., Huang, P., Liu, X., Tan, W., Yang, J.: Cascaded U-Net with training wheel attention module for change detection in satellite images. *Remote Sens.* **14**(24) (2022). <https://doi.org/10.3390/rs14246361>, <https://www.mdpi.com/2072-4292/14/24/6361>
2. Bandara, W.G.C., Patel, V.M.: A Transformer-based Siamese network for change detection. In: *IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 207–210 (2022). <https://doi.org/10.1109/IGARSS46834.2022.9883686>
3. Chen, H., Qi, Z., Shi, Z.: Remote sensing image change detection with Transformers. *IEEE Trans. Geosci. Remote Sens.* **60**, 1–14 (2022). <https://doi.org/10.1109/TGRS.2021.3095166>
4. Chen, H., Shi, Z.: A spatial-temporal attention-based method and a new dataset for remote sensing image change detection. *Remote Sens.* **12**, 1662 (2020). <https://doi.org/10.3390/rs12101662>
5. Chen, S., Yang, K., Stiefelbogen, R.: DR-TANet: dynamic receptive temporal attention network for street scene change detection. In: *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 502–509 (2021). <https://doi.org/10.1109/IV48863.2021.9575362>
6. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807 (2016). <https://api.semanticscholar.org/CorpusID:2375110>
7. Codegoni, A., Lombardi, G., Ferrari, A.: TINYCD: a (not so) deep learning model for change detection. *Neural Comput. Appl.* **35**, 8471–8486 (2023). <https://doi.org/10.1007/s00521-022-08122-3>

8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009). <https://doi.org/10.1109/CVPR.2009.5206848>
9. Du, P., Liu, S., Gamba, P., Tan, K., Xia, J.: Fusion of difference images for change detection over urban areas. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **5**(4), 1076–1086 (2012). <https://doi.org/10.1109/JSTARS.2012.2200879>
10. Feng, Y., Jiang, J., Xu, H., Zheng, J.: Change detection on remote sensing images using dual-branch multilevel intertemporal network. *IEEE Trans. Geosci. Remote Sens.* **61**, 1–15 (2023). <https://doi.org/10.1109/TGRS.2023.3241257>
11. Gabor, D.: Theory of communication. *J. Inst. Electr. Eng.* **93**(3), 429–457 (1946)
12. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018). <https://doi.org/10.1109/CVPR.2018.00745>
13. Huo, S., Zhou, Y., Zhang, L., Feng, Y., Xiang, W., Kung, S.Y.: Geometric variation adaptive network for remote sensing image change detection. *IEEE Trans. Geosci. Remote Sens.* **62**, 1–14 (2024). <https://doi.org/10.1109/TGRS.2024.3363431>
14. Ji, S., Wei, S., Lu, M.: Fully convolutional networks for multisource building extraction from an open aerial and satellite imagery data set. *IEEE Trans. Geosci. Remote Sens.* **57**(1), 574–586 (2019). <https://doi.org/10.1109/TGRS.2018.2858817>
15. Jiang, M., Chen, Y., Dong, Z., Liu, X., Zhang, X., Zhang, H.: Multiscale fusion CNN-Transformer network for high-resolution remote sensing image change detection. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **17**, 5280–5293 (2024). <https://doi.org/10.1109/JSTARS.2024.3361507>
16. Liu, C., Wechsler, H.: Gabor feature based classification using the enhanced Fisher linear discriminant model for face recognition. *IEEE Trans. Image Process.* **11**(4), 467–476 (2002). <https://doi.org/10.1109/TIP.2002.999679>
17. Luan, S., Chen, C., Zhang, B., Han, J., Liu, J.: Gabor convolutional networks. *IEEE Trans. Image Process.* **27**(9), 4357–4366 (2018). <https://doi.org/10.1109/TIP.2018.2835143>
18. Pei, G., Zhang, L.: Feature hierarchical differentiation for remote sensing image change detection. *IEEE Geosci. Remote Sens. Lett.* **19**, 1–5 (2022). <https://doi.org/10.1109/LGRS.2022.3193502>
19. Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., Dosovitskiy, A.: Do vision Transformers see like convolutional neural networks? In: Neural Information Processing Systems (2021). <https://api.semanticscholar.org/CorpusID:237213700>
20. Seydi, S.T., Hasanlou, M., Chanussot, J., Ghamisi, P.: BDD-Net+: a building damage detection framework based on modified Coat-Net. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **16**, 4232–4247 (2023). <https://doi.org/10.1109/JSTARS.2023.3267847>
21. Shen, L., et al.: S2Looking: a satellite side-looking dataset for building change detection. *Remote Sens.* **13**(24) (2021). <https://doi.org/10.3390/rs13245094>, <https://www.mdpi.com/2072-4292/13/24/5094>
22. Sun, Y., Zhang, X., Huang, J., Wang, H., Xin, Q.: Fine-grained building change detection from very high-spatial-resolution remote sensing images based on deep multitask learning. *IEEE Geosci. Remote Sens. Lett.* **19**, 1–5 (2022). <https://doi.org/10.1109/LGRS.2020.3018858>
23. Tan, M., Le, Q.: EfficientNet: rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning

- Research, vol. 97, pp. 6105–6114. PMLR (2019). <https://proceedings.mlr.press/v97/tan19a.html>
24. Wang, G.H., Gao, B.B., Wang, C.: How to reduce change detection to semantic segmentation. *Pattern Recogn.* **138**, 109384 (2023) <https://doi.org/10.1016/j.patcog.2023.109384>, <https://www.sciencedirect.com/science/article/pii/S0031320323000857>
 25. Wiskott, L., Fellous, J.M., Krüger, N., von der Malsburg, C.: Face recognition by elastic bunch graph matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**, 355–396 (1999). https://doi.org/10.1007/3-540-63460-6_150
 26. Ye, Y., Wang, M., Zhou, L., Lei, G., Fan, J., Qin, Y.: Adjacent-level feature cross-fusion with 3-D CNN for remote sensing image change detection. *IEEE Trans. Geosci. Remote Sens.* **61**, 1–14 (2023). <https://doi.org/10.1109/TGRS.2023.3305499>
 27. Zhai, X., Kolesnikov, A., Houlsby, N., Beyer, L.: Scaling vision Transformers. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12104–12113 (2022)



Task Consistent Prototype Learning for Incremental Few-Shot Semantic Segmentation

Wenbo Xu¹(✉), Yanan Wu², Haoran Jiang¹, Yang Wang³, Qiang Wu¹,
and Jian Zhang¹

¹ Faculty of Engineering and Information Technology, University of Technology
Sydney, Sydney 2007, Australia
Wenbo.xu@student.uts.edu.au

² School of Computer and Information Technology, Beijing Jiaotong University,
Beijing 100044, China

³ Department of Computer Science and Software Engineering, Concordia University,
Montreal H3G2J1, Canada

Abstract. Incremental Few-Shot Semantic Segmentation (iFSS) tackles a task that requires a model to continually expand its segmentation capability on novel classes using only a few annotated examples. Typical incremental approaches encounter a challenge that the objective of the base training phase (fitting base classes with sufficient instances) does not align with the incremental learning phase (rapidly adapting to new classes with less forgetting). This disconnect can result in suboptimal performance in the incremental setting. This study introduces a meta-learning-based prototype approach that encourages the model to learn how to adapt quickly while preserving previous knowledge. Concretely, we mimic the incremental evaluation protocol during the base training session by sampling a sequence of pseudo-incremental tasks. Each task in the simulated sequence is trained using a meta-objective to enable rapid adaptation without forgetting. To enhance discrimination among class prototypes, we introduce prototype space redistribution learning, which dynamically updates class prototypes to establish optimal inter-prototype boundaries within the prototype space. Extensive experiments on iFSS datasets built upon PASCAL and COCO benchmarks show the advanced performance of the proposed approach, offering valuable insights for addressing iFSS challenges.

Keywords: Few-shot segmentation · Prototype learning · Incremental learning · Meta-learning

1 Introduction

Deep learning models have made remarkable strides in semantic segmentation tasks by training on extensive datasets with rich annotations. In an effort

to alleviate the burden of data annotation, Few-shot Semantic Segmentation (FSS) [12,30] has been introduced, aiming to rapidly adapt to novel classes with minimal labeled data rapidly. However, FSS frameworks typically operate under a fixed output space assumption, where the number of target classes is predetermined. This limitation constrains the practicality and scalability of deployment in real-world scenarios where the total number of categories is uncertain, and new class objects may emerge over time.

In this work, we address a more challenging and practical scenario where the model continuously encounters a stream of new image data containing instances of previously unseen classes. The objective is to update a model to effectively segment new classes using a few annotated samples while retaining its segmentation capability on existing seen classes. The task known as Incremental Few-Shot Semantic Segmentation (iFSS) in the existing literature [2,33], is inspired by few-shot class incremental learning (FSCIL) [22]. It shares two common challenges, namely catastrophic forgetting of learned knowledge and overfitting to a limited number of novel class examples. This arises due to the absence of access to previous session data during the incremental learning sessions. When updating parameters with imbalanced novel class data (where the number of novel classes is considerably smaller compared to base classes), the model tends to exhibit a strong bias towards novel classes in pursuit of rapid adaptation. Consequently, there is a risk of aggressively overwriting crucial knowledge related to old classes in an attempt to accommodate the latest instances, resulting in a loss of generalization ability.

The challenges mentioned above stem from the task misalignment inherent in existing iFSS methods [2,33]. These methods begin by initializing a model that effectively predicts the base classes through classical supervised learning during the base training session. However, in subsequent incremental sessions, the focus shifts to pursuing fast adaptation to novel classes with less forgetting. To overcome this drawback, we propose a meta-learning [5,25] based approach that directly learns to incrementally adapt to novel classes conditioned on a few examples. This is achieved by simulating the incremental few-shot scenario during base session training (Fig. 1). Concretely, we sample a sequence of pseudo incremental tasks from the base class dataset. For each pseudo task, the model performs fast adaptation with a few new class examples and updates itself. Then the meta loss is calculated by measuring the performance of the updated model on the test images of both the old and the new classes. The object of the meta loss is to incentivize the model to incrementally learn new classes while minimizing the forgetting of the old ones.

Recently, some FSCIL studies [31,35] trains a backbone network on the base session and subsequently keep its parameters fixed during incremental sessions to maintain a consistent feature extractor. However, in these methods, the feature extractor remains static, implying that the feature space distributed for the base class is reused to accommodate additional classes. Our approach relies on prototype learning, wherein a prototype for a novel class is constructed from its features, forming a prototype classifier alongside the prototypes of the base

classes. When generating a prototype for a new class, it may be positioned close to the prototypes of the base classes in the feature space. This can result in interference, where a pixel will produce high similarity scores with both new and old prototypes, leading to catastrophic forgetting.

To optimize the prototype generation process, we propose a Prototype Space Re-distribution Learning (PSRL) to incrementally learn novel class prototypes and adaptively allocate base and novel prototypes into a latent prototype space, maintaining optimal prototype boundaries. Specifically, we fix the pre-trained feature backbone to preserve a unified feature extractor and introduce a prototype projector mapping intermediate class vectors to a subspace for dynamic prototype distribution. The redistribution process aims to enhance discrimination between new and old class prototypes, thereby improving segmentation performance. Furthermore, it regulates the updated base prototypes placed near their previous position to prevent prototype misalignment, effectively mitigating knowledge forgetting. The contributions of this work are summarized as:

- We introduce a meta-learning approach that closely aligns the base learning objective with the incremental evaluation protocol. Through training with a series of pseudo incremental tasks, our method directly optimizes the model to enhance the discovery of novel objects while mitigating forgetting
- We present Prototype Space Re-distribution Learning (PSRL), a method that incrementally learns novel classes while considering inter-prototype discrimination and maintaining base prototype consistency. This approach alleviates catastrophic forgetting of base classes and facilitates rapid adaptation to novel classes.
- Extensive experiments on dedicated iFSS benchmark from PASCAL and COCO datasets demonstrate the proposed method outperforms several counterparts.

2 Related Work

2.1 Semantic Segmentation

Semantic segmentation has witnessed significant advancements through the development of deep learning models. Fully Convolutional Networks (FCNs) [15] marked a significant milestone, enabling end-to-end learning for pixel-wise classification. Building upon this foundation, numerous Convolutional Neural Networks (CNNs) based architectures have been designed in aspects of optimal encoder-decoder frameworks [3], pyramid pooling [32] and multi-scale feature fusion [11]. While CNNs progressively build context through layers, transformers inherently consider the entire image at each stage, allowing them to capture long-range dependencies more effectively. This has led to the development of transformer-based models that introduce strong feature representation [24], hybrid CNN-Transformer architectures [29], and cross-attention decoders [4, 21]. Despite their impressive performance, these models typically require a substantial amount of mask-annotated data for training and are limited to predefined categories.

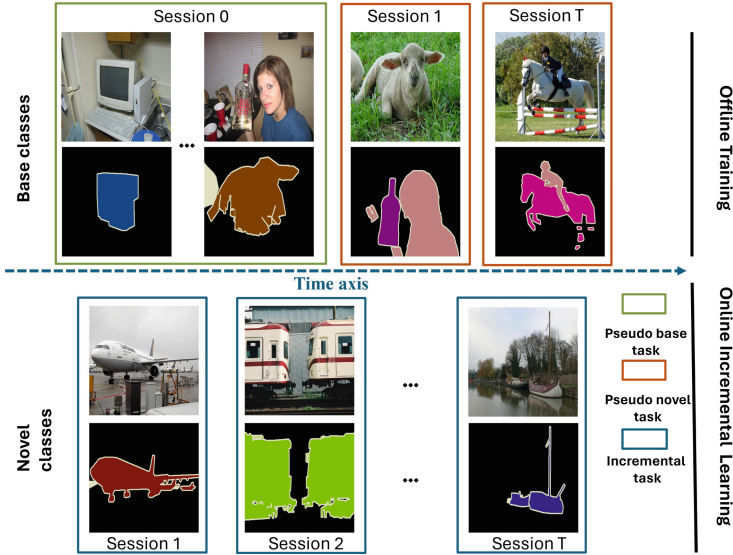


Fig. 1. Illustration of the evaluation protocol and our meta-training process. During the online incremental learning stage, the model undergoes training solely on new classes within each incremental session, while evaluation is conducted on all classes encountered thus far. Our strategy aims to replicate this evaluation protocol during the offline base class training stage. This is accomplished by randomly sampling a large portion of base class images to constitute the pseudo base dataset, with the remaining classes forming the pseudo novel classes. Initially, the model trains on the pseudo base dataset and subsequently adapts to the pseudo novel classes. This approach enables the model to learn how to swiftly identify new classes while retaining the ability to segment previously encountered ones

2.2 Few Shot Semantic Segmentation

To reduce the expenses associated with annotating segmentation data, researchers introduced few-shot semantic image segmentation (FSS). This approach aims to accurately segment objects in an image using only a small number of labeled examples per class. Drawing inspiration from few-shot learning [20], FSS models employ a two-branch architecture where a support branch learns class-wise prototypes from a small set of labeled images (support images), and the query image is segmented by comparing each pixel to the support class prototypes. Recent advancements in FSS mainly design models from the aspects of generating versatile prototypes and learning reliable feature correspondence. The prototype optimization strategy [13, 30] aims at compressing abstract class information into one or multiple prototypes that enable the model to perform effective feature guidance. The latter encourages the model to consider the most related information between the query-support images during segmentation by learning dense feature correspondence [16, 17]. Despite the progress made in few-shot semantic segmentation (FSS) methods, they specialize pri-

marily in identifying a single novel class by generating a binary foreground-background mask. In contrast, our prototype-based model addresses a more demanding and practical scenario where the model must segment all classes it has seen thus far.

2.3 Incremental Learning

Incremental learning (IL), also known as lifelong or continuous learning, is an approach within machine learning that focuses on the model’s ability to learn continuously, accommodating new knowledge while retaining previously learned information. IL methods can be broadly classified into replay-based [26] and regularization-based [34]. In replay-based methods, samples of previous tasks are either stored or generated at first and then replayed when learning the new task. Zhu et al. [34] propose to store the same number of old samples as each new class to form a joint set during its incremental learning process. Regularization-based methods protect old knowledge from being covered by imposing constraints on new tasks. In iFSS, Cermelli et al. [2] introduced a prototype-based distillation loss to force the current model to retain scores for old classes, thereby preventing forgetting. Guangchen et al. [19] proposes an embedding adaptive-update strategy to prevent catastrophic forgetting, where hyper-class embeddings remain fixed to preserve existing knowledge. To mitigate feature embedding bias, Kai et al. [10] presents class-agnostic foreground perception across multiple targets. Different from those methods, our method exploits the prototype classifier to remember knowledge and directly optimize the learning process with meta-learning tasks.

3 Method

3.1 Problem Setting

iFSS addresses the challenge of updating a pre-trained segmentation model to accommodate newly introduced classes over time, utilizing limited annotated examples for each novel class. Specifically, let $\mathcal{D}_{train/test}^t = \{\mathcal{I}_n^t, \mathcal{M}_n^t\}$, $n \in \{1, 2, \dots, K\}$, $t \in \{1, 2, \dots, T\}$, denote a sequence of the training and testing sets of image $\mathcal{I}_{train/test}^t$ and their corresponding semantic label masks $\mathcal{M}_{train/test}^t$. The label classes \mathcal{C}^t of each set are disjoint, such that $\mathcal{C}^i \cap \mathcal{C}^j = \emptyset, \forall i \neq j$. iFSS comprises a base session with abundant labeled training images from D_{train}^0 and a sequence of incremental sessions with only a few training images for each novel class from $\{D_{train}^1, D_{train}^2, \dots, D_{train}^T\}$. We undertake offline training in the base session to initialize a model using base classes \mathcal{C}^0 . After the base session, the model is expected to adapt to new classes $\mathcal{C}^t (t > 0)$ with a few examples in the subsequent incremental sessions. Note that at the t^{th} session, the model has access only to D_{train}^t for training and then is evaluated on test images containing all the encountered classes so far, i.e. $\{D_{test}^0 \cup D_{test}^1 \dots \cup D_{test}^t\}$.

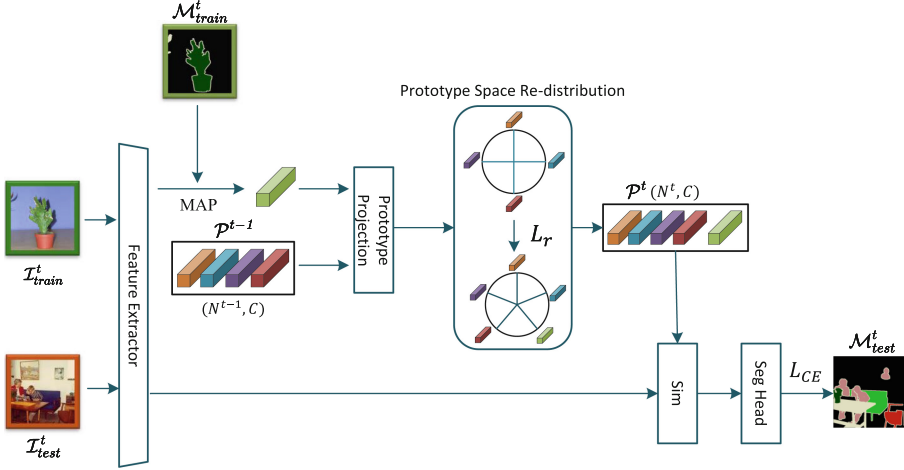


Fig. 2. The proposed prototype-based approach utilizes masked average pooling (MAP) to derive the novel class prototype. Subsequently, all prototypes are projected into a latent prototype space for redistribution. The resulting prototypes form a new classifier \mathcal{P}^t capable of identifying both base and novel classes. This process is considered as a sequential task of the meta-learning optimization. In the online incremental sessions, the feature extractor remains frozen, and only the prototype projector and segmentation head are updated

3.2 Prototype Space Re-distribution Learning

Prototype-Based Semantic Segmentation. As introduced in [23], typical prototype-based few-shot semantic segmentation frameworks comprise a feature extractor and a prototype classifier. Feature extractor transforms the input image $\mathcal{I} \in \mathbb{R}^{h \times w \times 3}$ into a feature embedding $\mathcal{F} \in \mathbb{R}^{w \times h \times d}$ in a latent space. Subsequently, a prototype classifier $\mathcal{P} \in \mathcal{R}^{N \times d}$ is trained to perform pixel-wise predictions for N classes on \mathcal{F} . For iFSS, our objective is to progressively expand the base prototype classifier \mathcal{P}^0 with prototypes of novel classes, facilitating the continuous segmentation of newly encountered classes without forgetting prior knowledge. Formally, in an N -class K -shot incremental session (N novel classes and each novel class has K training samples), all training samples $\mathcal{I}_{c,n}^t$ are first processed by a feature extractor f and mask average pooling. Subsequently, these samples are averaged over K shots to create N prototypes, denoted as $p_c^t (c \in \{1, 2, \dots, N\})$.

$$p_c^t = \frac{1}{K} \sum_{n=1}^K \frac{\sum_{h,w} [\mathcal{M}_{c,n}^t \circ f(\mathcal{I}_{c,n}^t)]_{h,w}}{\sum_{h,w} [\mathcal{M}_{c,n}^t]_{h,w}}, \quad (1)$$

where $\mathcal{I}_{c,n}^t$ denotes the n -th training image of class \mathbf{c} . $\mathcal{M}_{c,n}^t \in \mathbb{R}^{h,w,1}$ is the class mask for class \mathbf{c} on feature $f(\mathcal{I}_{c,n}^t) \in \mathbb{R}^{h,w,d}$. After obtaining N prototypes, the prediction of pixel i of \mathcal{F} is assigned according to the normalized cosine similarity score $S_{i,c}(\mathcal{F})$ between features and the class prototype p_c^t as:

$$S_{i,c}(\mathcal{F}) = \frac{\exp\left(\text{Sim}(\mathcal{F}_i, \mathbf{p}_c^t)/\tau\right)}{\sum_{j=1}^N \exp\left(\text{Sim}(\mathcal{F}_i, \mathbf{p}_j^t)/\tau\right)}, \quad (2)$$

where $\mathcal{F}_i \in \mathbb{R}^d$ are the positional features extracted from input image \mathcal{I} , N represents the cumulative category of prototype vectors up to session t , and τ is a temperature parameter that controls the concentration level of the distribution [27]. $\text{Sim}(\cdot) = \frac{\mathcal{F}_i^\top \mathbf{p}^t}{\|\mathcal{F}_i\| \|\mathbf{p}^t\|}$ is the cosine similarity metric that measures the pixel classification score.

Training with few-shot examples in novel class sessions inevitably leads to overfitting and has the potential to undermine the feature extraction capabilities of the pre-train backbone network. Given that our prototype classifier encompasses both base and newly encountered classes, and we have no access to base examples during incremental learning, modulating the extractor may map new classes into a disparate feature space from that of base classes. Therefore, to ensure consistent feature mapping, the backbone is consistently maintained in a fixed state. However, the newly added prototypes may lie close to the base-class prototypes because the prototype is derived from a fixed feature space that is tailored for base classes. To discriminate novel prototypes from their base counterparts, we introduce the prototype projector \mathbf{g} to map the current prototypes into a latent prototype space (Fig. 2) where base and novel prototypes are adaptively distributed to achieve two objectives: i) ensuring clear inter-prototype discrimination among base and novel prototypes for fast adaptation to new classes, and ii) minimizing the displacement of base prototypes away from their original positions to prevent catastrophic forgetting and maintain alignment between features and prototypes. Accordingly, we propose a novel prototype redistribution loss that places the new class prototype p_i^t at a position far from base prototypes P_j^{t-1} and relocates base classes to a near-optimal position as:

$$\mathcal{L}_r = \frac{\sum_{i=1}^{N^b} \sum_{j=1}^{N^t} \text{Sim}(P_i^{t-1}, P_j^t)}{\sum_{i=1}^{N^b} \text{Sim}(P_i^{t-1}, \hat{P}_i^{t-1})}, \quad (3)$$

where N^b, N^t are the class prototype number of previous sessions $[0, 1, \dots, t-1]$ and current session t . \hat{P}_i^{t-1} represents the redistributed prototype vector derived from the base prototype P_i^{t-1} . We utilize cosine distance as the metric for the similarity matrix. The loss function \mathcal{L}_r is designed to minimize the similarity between new class prototypes and base prototypes, concurrently maximizing the similarity between the original base prototypes and their respective redistributions.

3.3 Learning to Incrementally Learn

The core idea underlying our approach is meta-learning inspired by MAML [7] for few-shot tasks. During the meta-training phase, the model is trained with a set of

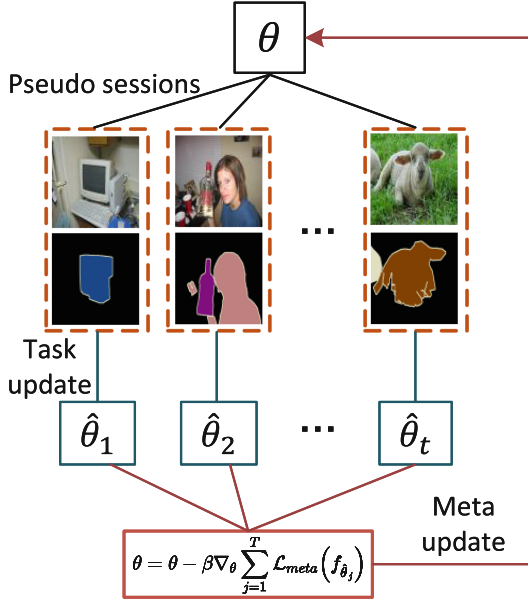


Fig. 3. The meta-learning optimization strategy samples pseudo-sequential learning tasks on the base set to perform task training. The meta update process encourages the model to learn in a manner that preserves performance on old classes while effectively adapting to novel classes.

novel class adaptation tasks that are formulated as few-shot learning problems, aiming to simulate the scenario encountered during meta-testing. In iFSS, the online incremental stage closely resembles the “meta-testing” stage. This stage entails adapting the model to a sequence of incremental sessions, where each session introduces several novel classes with few-shot examples. Inspired by this, the model is meta-trained on base classes with the goal of mimicking the incremental learning scenario anticipated during the subsequent online incremental learning (i.e., evaluation). This ensures that the model is learned in a manner enabling effective adaptation to new classes with less forgetting.

Sequential Task Sampling. We replicate the evaluation process by utilizing the base classes. More precisely, we segregate the training images of base classes into distinct training and testing sets with no overlap. In each epoch, we initiate the training process by sampling a sequence of T tasks, $\mathcal{D}_{\text{train/test}}^s = \left\{ \left(\mathcal{I}_{\text{train/test}}^j, \mathcal{M}_{\text{train/test}}^j \right) \right\}_{j=0}^T$, where T is the actual incremental session number, and each session include training and testing image-mask pairs. We define D^0 as the pseudo base set, comprising more classes and training examples than subsequent tasks (e.g., $j > 0$) in the few-shot setting. To mitigate the risk of the model overfitting to a particular images, we introduce random sampling of classes and their corresponding images

Algorithm 1. Meta training process**Require:** θ^g, θ^{seg} : pre-trained weights**Require:** \mathcal{D}^0 : training set of base classes

```

1: while not converged do
2:    $\mathcal{D}_{train/test}^s = \left\{ \left( \mathcal{I}_{train/test}^j, \mathcal{M}_{train/test}^j \right) \right\}_{j=0}^T$ 
3:   Initialize the models with  $\mathcal{D}^0$ .
4:    $\mathcal{D}_{meta} = \emptyset$ 
5:   for  $j = 1, 2, \dots, T$  do
6:      $\mathcal{P} = \text{Concat}(\mathcal{P}_{old}, \mathcal{P}_{new})$ 
7:      $\hat{\theta}_j^{g,seg} = \theta^{g,seg} - \alpha \nabla_{\theta^{g,seg}} \mathcal{L}_{CE} \left( \mathcal{I}_{train}^j, \mathcal{M}_{train}^j; \theta \right)$ 
8:      $\mathcal{D}_{meta} = \mathcal{D}_{meta} \cup \mathcal{D}_{test}^j$ 
9:      $\theta^{g,seg} = \theta^{g,seg} - \beta \nabla_{\theta^{g,seg}} \sum_{(\mathcal{I}, \mathcal{M}) \in \mathcal{D}_{meta}} \mathcal{L}_{meta} \left( \mathcal{I}, \mathcal{M}; \hat{\theta}_j^{g,seg} \right)$ 
10:   end for
11: end while

```

Meta-training. During the meta-training phase, for every sampled sequence $\mathcal{D}_{train/test}^s$, we introduce a prototype redistribution-oriented optimization approach grounded in Meta-Learning. We denote $\theta = \{\theta^f, \theta^g, \theta^{seg}\}$ as the parameter for the whole network, where $\theta^f, \theta^g, \theta^{seg}$ denote the parameters for backbone, prototype projection layer and segmentation head, respectively. We first conduct supervised training of θ on the base classes using cross-entropy loss (\mathcal{L}_{CE}). The meta-training procedure is illustrated in Algorithm 1 and Fig. 3. At the beginning of training on each sequence, we define an empty cumulative meta test set \mathcal{D}_{meta} to store the test images from previous tasks. At the j^{th} task, we first generate the new class prototypes \mathcal{P}_{new} and then concatenate it into the current prototype classifier \mathcal{P}_{old} . Subsequently, we start to perform fast adaptation to new classes and update θ^g and θ^{seg} via a few L gradient steps:

$$\hat{\theta}_j^{g,seg} = \theta^{g,seg} - \alpha \nabla_{\theta^{g,seg}} \mathcal{L}_{CE} \left(\mathcal{I}_{train}^j, \mathcal{M}_{train}^j; \theta \right), \quad (4)$$

where $\mathcal{I}_{train}^j, \mathcal{M}_{train}^j$ are the images and labels for training j^{th} pseudo task. The loss $\mathcal{L}_{CE}(\cdot, \cdot)$ is computed on the output of the current model and the target label \mathcal{M}_{train}^j .

The adaptation process mimics the model’s learning pattern for new classes during incremental sessions. Ideally, we aim for the adapted parameters to perform well in both the classes from the previous and current tasks. The meta-test set accumulated from previous tasks is used for evaluating how well the updated model resists catastrophic forgetting on old classes and adaptation on new classes. We append \mathcal{D}_{test}^j to \mathcal{D}_{meta} , and accordingly, the meta-objective is defined as:

$$\theta^{g,seg} = \theta^{g,seg} - \beta \nabla_{\theta^{g,seg}} \sum_{(\mathcal{I}, \mathcal{M}) \in \mathcal{D}_{meta}} \mathcal{L}_{meta} \left(\mathcal{I}, \mathcal{M}; \hat{\theta}_j^{g,seg} \right). \quad (5)$$

Note that \mathcal{L}_{meta} is a function designed to optimize $\theta^{g,seg}$ with the objective of achieving optimal performance through the redistribution of class prototypes as:

$$\mathcal{L}_{meta} = \mathcal{L}_{CE}(\mathcal{I}_{test}, \mathcal{M}_{test}) + \lambda \mathcal{L}_r. \quad (6)$$

When all N tasks are done, \mathcal{D}_{meta} is reset to empty and we repeat the learning process from the random initialization and adaptation process.

In the online incremental learning stage, we execute Lines 5–7 of Algorithm 1 to acquire knowledge about novel classes during evaluation. The steps outlined in Algorithm 1 align with the evaluation protocol: after being trained on the current session, the model undergoes evaluation on all encountered classes so far. This meta-objective encourages our model to quickly adapt to novel classes without sacrificing remembering old ones.

4 Experiments

4.1 Dataset

We evaluate the proposed method on two widely used semantic segmentation datasets: PASCAL VOC 2012 [6] and COCO [14]. Following established practices [2], we evenly partition the classes in PASCAL VOC and COCO into four folds, with each fold containing 5 and 20 categories, respectively. In the validation stage, three folds are used to form the base set, while the categories from the remaining fold are utilized for testing.

4.2 Implementation Details and Evaluation Metrics

In all experiments, we employ ResNet-101 [9] pre-trained on ImageNet as the feature extractor. Our configuration involves ASPP [3] with a 1×1 convolutional layer as the segmentation head. We evaluate the performance of a method utilizing three mean intersection-over-union (mIoU) metrics: mIoU on base classes (mIoU-B), mIoU on new classes (mIoU-N), and the harmonic mean of the two (HM). Consistent with [2], all reported results are presented upon the completion of training in the final incremental session. Particularly, the single step means while multi-step has multiple sessions: 5 sessions of 1 class on VOC and 4 sessions of 5 classes on COCO.

4.3 Main Results

The outcomes of our method on the PASCAL VOC 2012 and COCO datasets are consolidated in Table 1 and Table 2, respectively. We consider three baselines: Finetune, directly fine-tune the base model with new classes on each session; naive prototype classifier WI [18] and its dynamic version DWI [8]; knowledge-distillation-based method MiB [1]; FSS method HDMNet [17] and iFSS approach SRAA [33]. Our approach demonstrates superior performance in novel class adaptation across most settings for both PASCAL and COCO datasets.

Table 1. The experimental results on the PASCAL VOC 2012 dataset.

Method	Single step						Multi-step					
	1-shot			5-shot			1-shot			5-shot		
	mIoU-B	mIoU-N	HM	mIoU-B	mIoU-N	HM	mIoU-B	mIoU-N	HM	mIoU-B	mIoU-N	HM
Finetune	58.3	9.7	16.7	55.8	29.6	38.7	47.2	3.9	7.2	58.7	7.7	13.6
WI [18]	62.7	15.5	24.9	64.9	21.7	32.5	66.6	16.1	25.9	66.6	21.9	33.0
DWI [8]	64.3	15.4	24.8	64.9	23.5	34.5	67.2	16.3	26.2	67.6	25.4	36.9
MiB [1]	61.0	5.2	9.7	65.0	28.1	39.3	43.9	2.6	4.9	60.9	5.8	10.5
SPN [28]	59.8	16.3	25.6	58.4	33.4	42.5	49.8	8.1	13.9	61.6	16.3	25.8
PIFS [2]	60.9	18.6	28.5	60.5	33.4	43.0	64.1	16.9	26.7	64.5	27.5	38.6
HDMNet [17]	57.7	16.4	29.5	58.1	34.9	43.6	52.2	15.6	19.0	55.0	14.7	23.2
SRAA [33]	65.2	19.1	25.5	63.8	36.7	46.6	66.4	18.8	29.3	64.3	28.7	39.7
Ours	63.4	19.7	30.1	61.6	35.8	45.3	65.5	20.4	31.1	65.9	29.1	40.4

Table 2. The experimental results on the COCO dataset.

Method	Single step						Multi-step					
	1-shot			5-shot			1-shot			5-shot		
	mIoU-B	mIoU-N	HM	mIoU-B	mIoU-N	HM	mIoU-B	mIoU-N	HM	mIoU-B	mIoU-N	HM
Finetune	41.2	4.1	7.5	41.6	12.3	19.0	38.5	4.8	8.5	39.5	11.5	17.8
WI [18]	43.8	6.9	11.9	43.6	8.7	14.5	46.3	8.3	14.1	46.3	10.3	16.9
DWI [8]	44.5	7.5	12.8	44.9	12.1	19.1	46.2	9.2	15.3	46.6	14.5	22.1
MiB [1]	43.8	3.5	6.5	44.7	11.9	18.8	40.4	3.1	5.8	43.8	11.5	18.2
SPN [28]	43.5	6.7	11.7	43.7	15.6	22.9	40.3	8.7	14.3	41.4	18.2	25.3
PIFS [2]	40.8	8.2	9.8	41.4	9.6	15.6	39.7	5.9	10.3	40.3	16.3	23.2
HDMNet [17]	39.5	5.6	9.8	40.1	13.6	20.3	39.7	6.5	11.2	41.4	12.6	19.3
SRAA [33]	41.2	9.3	15.2	46.2	17.1	24.4	40.7	11.3	17.7	41.0	19.7	26.6
Ours	43.8	10.4	16.7	44.4	20.8	28.3	43.1	12.3	19.1	43.5	22.2	29.4

Additionally, it achieves state-of-the-art performance in terms of Harmonic Mean (HM) scores across all settings except for 5-shot single step, indicating that our approach effectively balances the retention of information about old classes while facilitating adaptation to new ones. Particularly noteworthy is our method’s performance on the PASCAL dataset, where it achieves significantly higher novel class segmentation mIoU scores compared to all other methods, reaching 35.8% and 29.1% in single-step and multi-step settings, respectively. This surpasses the state-of-the-art method (SRAA) by 0.6% and 1.8% under 1-shot setting, respectively. Our meta-learning-based approach exhibits superior fast adaptation capability to novel classes without compromising base class segmentation accuracy, achieving competitive base class segmentation performance on both PASCAL and COCO datasets. In the single-step setting, all the new classes are introduced in a single session. When more samples are provided for a particular class, the model demonstrates improved adaptation to the novel class, as evidenced by the mIoU-N score, albeit with a potential decrease in performance for the base classes. This effect is mitigated in the multi-step setting, where our meta-learning approach effectively learns to resist forgetting through training across multiple sessions

On the COCO dataset, our approach showcases significantly greater improvements in HM scores compared to the state-of-the-art method SRAA [33]. For instance, in the task of 5-shot segmentation, our method’s HM scores surpass those of SRAA by 3.9% and 2.8%, whereas the margins are only -1.3% and 0.7%

on the PASCAL dataset. This highlights the effectiveness of our approach in tackling the more intricate challenges associated with a larger number of classes, which is particularly beneficial in real-world applications.

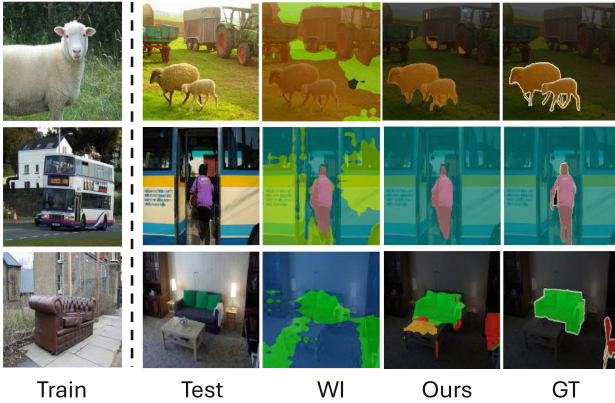


Fig. 4. Visualization of multi-step results under shot setting on the PASCAL dataset.

In Fig. 4, we showcase visualized segmentation results obtained from training under the multi-step incremental setup, using one training example for each novel class. In comparison to vanilla weight-printing (WI), which simply appends new class prototypes to the prototype classifier, our approach notably distinguishes novel classes like “bus” from the base class “person” and “sheep” from the background. Additionally, as observed in the third row, WI exhibits overfitting to the “sofa” and completely forgets the knowledge of the “chair”. Our method, employing task-consistent meta-learning with prototype distribution loss, preserves the ability to segment learned classes while accurately adapting to new classes.

4.4 Ablation Study

Component Effectiveness. As illustrated in the second row of Table 3, introducing the meta-learning strategy, which trains the model in a manner aligned with the expected evaluation in the incremental sessions, significantly enhances novel class adaptation and mitigates catastrophic forgetting. The application of \mathcal{L}_{inter} upon meta-learning results in a 0.9% increase in novel class accuracy but induces a 1.3% performance reduction in the base class. It suggests that merely focusing on minimizing the similarity between the new class and the old class prototypes while neglecting the drift of the base class can lead to prototype inconsistency before and after adaptation, resulting in knowledge forgetting.

Backbone and Prototype Redistribution. To investigate the performance difference between frozen and updated backbones, we conduct comparison exper-

Table 3. Ablation study of the meta-learning scheme and prototype redistribution loss on COCO, under the multi-step one-shot setting. $\mathcal{L}_{\text{inter}} = \sum_{i=1}^{N^b} \sum_{j=1}^{N^t} \text{Sim}(P_i^{t-1}, P_j^t)$ merely aims to minimize similarity between novel and base classes

Baseline	Meta-learning	$\mathcal{L}_{\text{inter}}$	\mathcal{L}_r	Base	Novel	HM
✓				44.8	7.8	13.3
✓	✓			42.5	10.6	17.0
✓	✓	✓		41.2	11.5	18.0
✓	✓		✓	43.1	12.3	19.1

Table 4. Ablations on backbones and prototype redistribution. “fix” denotes that the backbone remains fixed during incremental steps, while “update” means that the backbone continues to update. “PR” indicates the addition of the prototype projection layer and the adoption of the prototype redistribution loss \mathcal{L}_r .

Methods	Novel	Base	HM
Backbone (fix)	7.2	44.1	12.4
Backbone (update)	7.8	36.0	12.8
Backbone (fix) + PR	10.6	40.4	16.8
Backbone (update) + PR	10.2	36.5	15.9

iments using two baseline models. In these experiments, the pre-trained backbone is either kept fixed or updated during the incremental steps. The model with the fixed backbone is denoted as Backbone (fix), while the model with the updated backbone is referred to as Backbone (update). As shown in Table 4, Backbone (update) outperforms Backbone (fix) in terms of HM score, primarily due to its superior performance on novel classes. However, there is a significant drop in mIoU for base classes, indicating that updating the backbone without any constrain may lead to overfitting on new classes and result in catastrophic forgetting.

Then, we augment the model by appending a prototype projection layer after the backbone and applying prototype redistribution supervision to obtain the classifier. From the results of the last two rows of Table 4, the fixed version outperforms the updated counterpart by a significant margin in both novel and base class segmentation. This superiority is attributed to the fixed backbone’s ability to retain information about the base classes, while “PR” ensures that the prototypes in the subspace remain well-separated. These factors mitigate catastrophic forgetting and facilitate rapid adaptation.

Coefficient Selection. As shown in Fig. 5, we investigate the impact of the coefficient λ in Eq. 6 on the model’s performance, testing lambda values from 0.1 to 0.6. The objective is to identify the optimal λ that balances regularization with the model’s ability to effectively learn new classes.

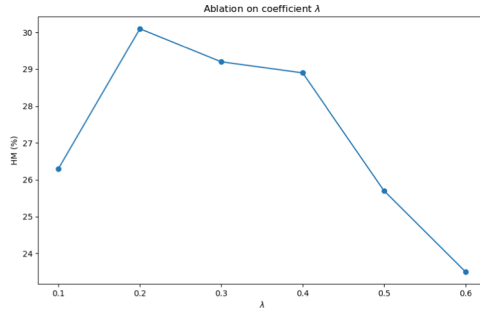


Fig. 5. Ablation study on coefficient λ . HM performance in the Single step experiment under 1-shot setting.

Our findings indicate that a λ value of 0.3 achieves the best performance, as evidenced by the peak in the performance of HM under 1-shot setting. This optimal performance at $\lambda = 0.3$ suggests an effective balance between forgetting and adaptability. Lower λ values, closer to 0.1, may result in insufficient regularization, causing overfitting and poor generalization. Conversely, higher λ values, approaching 0.6, could overly constrain the model, limiting its ability to adapt to novel classes and thereby degrading performance.

5 Conclusion

This work addresses a practical scenario of semantic segmentation that incrementally learns novel classes with a few examples. We propose a meta-learning-based approach, directly optimizing the network to acquire the ability to incrementally learn within the few-shot incremental setting. To alleviate catastrophic forgetting and overfitting problems, we introduce a prototype space re-distribution mechanism to dynamically update class prototypes during each incremental session. Extensive experiments on PASCAL and COCO benchmarks demonstrate that the proposed method facilitates a model learning paradigm for quick classes learning without forgetting.

References

1. Cermelli, F., Mancini, M., Bulo, S.R., Ricci, E., Caputo, B.: Modeling the background for incremental learning in semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9233–9242 (2020)
2. Cermelli, F., Mancini, M., Xian, Y., Akata, Z., Caputo, B.: Prototype-based incremental few-shot semantic segmentation. arXiv preprint [arXiv:2012.01415](https://arxiv.org/abs/2012.01415) (2020)
3. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 833–851. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_49

4. Cheng, B., Misra, I., Schwing, A.G., Kirillov, A., Girdhar, R.: Masked-attention transformer for universal image segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1290–1299 (2022)
5. Chi, Z., Gu, L., Liu, H., Wang, Y., Yu, Y., Tang, J.: MetaFSCIL: a meta-learning approach for few-shot class incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14166–14175 (2022)
6. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. *Int. J. Comput. Vision* **88**, 303–338 (2010)
7. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International Conference on Machine Learning, pp. 1126–1135. PMLR (2017)
8. Gidaris, S., Komodakis, N.: Dynamic few-shot visual learning without forgetting. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4367–4375 (2018)
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. Huang, K., Wang, F., Xi, Y., Gao, Y.: Prototypical kernel learning and open-set foreground perception for generalized few-shot semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 19256–19265 (2023)
11. Kirillov, A., Girshick, R., He, K., Dollár, P.: Panoptic feature pyramid networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6399–6408 (2019)
12. Lang, C., Cheng, G., Tu, B., Li, C., Han, J.: Base and meta: a new perspective on few-shot segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* (2023)
13. Li, G., Jampani, V., Sevilla-Lara, L., Sun, D., Kim, J., Kim, J.: Adaptive prototype learning and allocation for few-shot segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8334–8343 (2021)
14. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) *ECCV 2014, Part V. LNCS*, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
15. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(4), 640–651 (2015)
16. Min, J., Kang, D., Cho, M.: Hypercorrelation squeeze for few-shot segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6941–6952 (2021)
17. Peng, B., et al.: Hierarchical dense correlation distillation for few-shot segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 23641–23651 (2023)
18. Qi, H., Brown, M., Lowe, D.G.: Low-shot learning with imprinted weights. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5822–5830 (2018)
19. Shi, G., Wu, Y., Liu, J., Wan, S., Wang, W., Lu, T.: Incremental few-shot semantic segmentation via embedding adaptive-update and hyper-class representation. In: Proceedings of the 30th ACM International Conference on Multimedia, pp. 5547–5556 (2022)

20. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
21. Strudel, R., Garcia, R., Laptev, I., Schmid, C.: Segmenter: transformer for semantic segmentation. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7262–7272 (2021)
22. Tao, X., Hong, X., Chang, X., Dong, S., Wei, X., Gong, Y.: Few-shot class-incremental learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12183–12192 (2020)
23. Tian, Z., et al.: Generalized few-shot semantic segmentation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11563–11572 (2022)
24. Wang, W., et al.: Pyramid vision transformer: a versatile backbone for dense prediction without convolutions. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 568–578 (2021)
25. Wu, Y., Chi, Z., Wang, Y., Feng, S.: MetaGCD: learning to continually learn in generalized category discovery. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1655–1665 (2023)
26. Wu, Y., et al.: Large scale incremental learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 374–382 (2019)
27. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3733–3742 (2018)
28. Xian, Y., Choudhury, S., He, Y., Schiele, B., Akata, Z.: Semantic projection network for zero- and few-label semantic segmentation. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8256–8265 (2019)
29. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: SegFormer: simple and efficient design for semantic segmentation with transformers. *Adv. Neural. Inf. Process. Syst.* **34**, 12077–12090 (2021)
30. Xu, W., Huang, H., Cheng, M., Yu, L., Wu, Q., Zhang, J.: Masked cross-image encoding for few-shot segmentation. In: *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 744–749. IEEE (2023)
31. Zhang, C., Song, N., Lin, G., Zheng, Y., Pan, P., Xu, Y.: Few-shot incremental learning with continually evolved classifiers. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12455–12464 (2021)
32. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2881–2890 (2017)
33. Zhou, Y., Chen, X., Guo, Y., Yu, J., Hong, R., Tian, Q.: Advancing incremental few-shot semantic segmentation via semantic-guided relation alignment and adaptation. In: Rudinac, S., et al. (eds.) *MMM 2024. LNCS*, vol. 14554, pp. 244–257. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-53305-1_19
34. Zhu, J., Yao, G., Zhou, W., Zhang, G., Ping, W., Zhang, W.: Feature distribution distillation-based few shot class incremental learning. In: *2022 5th International Conference on Pattern Recognition and Artificial Intelligence (PRAI)*, pp. 108–113. IEEE (2022)
35. Zhu, K., Cao, Y., Zhai, W., Cheng, J., Zha, Z.J.: Self-promoted prototype refinement for few-shot class-incremental learning. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6801–6810 (2021)



Multi-scale Value-Density Transformer with Medical Semantic Guidance for Disease Risk Prediction Based on Clinical Time Series

Jingwen Xu, Xiaoge Wei, and Pong C. Yuen^(✉)

Department of Computer Science, Hong Kong Baptist University, Hong Kong, China
{csjwxu, xgwei, pcyuen}@comp.hkbu.edu.hk

Abstract. Clinical time series are ubiquitous in healthcare for accurate disease risk prediction. The recent Transformer models have demonstrated superior performance in time series learning. However, these methods focus on global temporal dependency and widely utilize channel-mix or channel-independent tokenization. They ignore local dependency in clinical time series and are limited to capturing the intrinsic clinical variable correlations. To address the above issues, we present an **Multi-scale Value-Density Transformer with Medical Semantic Guidance (MVMformer)**, which takes irregularity-aware segment-wise modeling for clinical time series and correlate diverse clinical variates based on their medical semantic affinity. Specifically, MVMformer introduces a Multi-scale Value-Density Attention to capture intra-segment characteristics in both value trends and temporal density while accommodating multi-length segments. Furthermore, MVMformer constructs a Hierarchical Medical Semantic Graph to analyze complicated variable relationships starting from detailed measurements to associated organs. Experimental results on three medical datasets demonstrate the superiority of MVMformer over existing state-of-the-art methods.

Keywords: Clinical Time Series · Segment · Kernelized · Intensity · Medical Semantic

1 Introduction

The widespread application of Electronic Health Records (EHRs) accumulated a large amount of clinical time series data, which consists of patients' examined clinical variates at multiple visits to hospitals [16, 26, 30]. Early disease prediction based on clinical time series data is crucial for patients [15, 31, 35], which supports clinicians with timely intervention. However, modeling clinical time series presents its unique challenges in two aspects: (1) *Temporal Dependency*: The temporal dependency in clinical time series involves the adaptive combination of both global dependency and local dependency. In particular, local dependency exhibits distinct characteristics (*e.g.*, different patient states within specific time

periods), which are significant aspects in clinical time series analysis. (2) *Clinical Variable Correlations*: The correlations between pairs of clinical variables can be complicated and vary in magnitude. For instance, there are strong correlations between urea and creatinine to indicate kidney dysfunction, compared to other variables.

The recent Transformer-based methods have demonstrated strong performance in clinical time series learning [3, 13, 28, 29, 32]. While these methods have addressed global dependency through effective self-attention design [12, 18], they tend to overlook the informative local dependency in clinical time series data. Furthermore, in terms of clinical variable correlations, existing approaches often employ channel-mixing or channel-independent tokenization [6, 19], assuming full correlation among clinical variables and leading to high learning complexity.

To tackle the above issues, our motivations are in two-fold: (1) *Irregularity-aware segment-wise modeling*: These segments contain valuable clues that reflect the patient states within specific time periods. However, modeling the segment in clinical time series can be challenging due to irregularity. As illustrated in Fig. 1, there are varying dominant segment lengths and significant intra-segment characteristics in both the value and time distributions. (2) *Connecting clinical variables based on medical semantic affinity*: Clinical variables contain specific medical meanings, which provide valuable semantic information. According to their medical meanings, we can establish correlations between clinical variables from various aspects. For instance, as illustrated in Fig. 1, it is commonly observed that clinical variables are sparsely associated with particular organs based on their medical meanings [17, 21, 23].

Based on the above analysis, a **Multi-scale Value-Density Transformer** with **Medical Semantic Guidance (MVMformer)** is proposed to perform disease prediction using clinical time series modeling. MVMformer takes irregularity-aware segment-wise modeling while connecting diverse clinical variables based on their medical semantics: To capture the intra-segment characteristics and achieve multi-scale modeling, we introduce the Multi-Scale Value-Density Attention (MS-VDAtn). MS-VDAtn accommodates multi-length segments in a single module and captures both value trend and temporal density within segment. To connect diverse clinical variables, a Hierarchical Medical Semantic Graph (HMS Graph) is proposed to explore the group similarity of clinical variables through their associated organs. The HMS Graph enables a comprehensive patient analysis starting from detailed measurements and extending to associated organs. Our extensive experiments on three real-world medical datasets demonstrate the effectiveness of MVMformer against state-of-the-art models. The main contributions are summarized as follows:

- We introduce a **Multi-scale Value-Density Transformer** with **Medical Semantic Guidance (MVMformer)** for disease prediction based on clinical time series. Our method focuses on capturing local dependency through irregularity-aware segment-wise modeling and complicated clinical variables based on their medical semantics.

- A Multi-Scale Value-Density Attention (MS-VDAtn) is proposed to handle multi-length segments and capture intra-segment characteristics in both value trend and temporal density.
- A Hierarchical Medical Semantic Graph (HMS Graph) is proposed to establish connections between clinical variables from related measurements to associated organs.
- Experimental results on three real-world clinical time series datasets show that our method significantly outperforms all baseline methods.

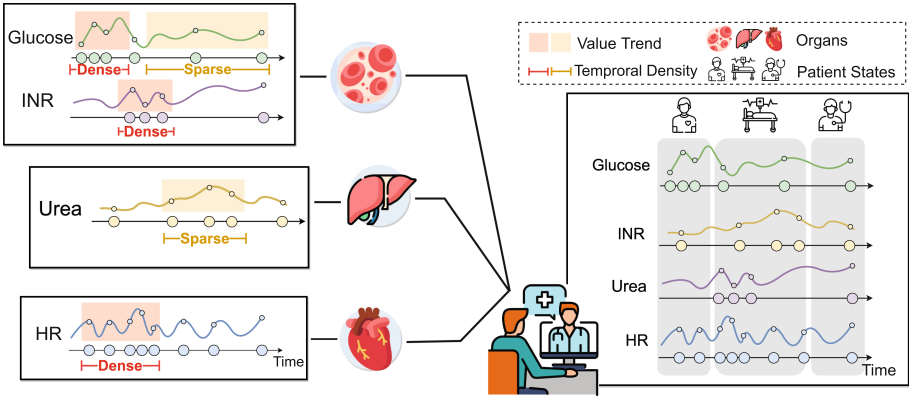


Fig. 1. Clinical time series exhibit significant local dependency in both value trend and temporal density, attributed to distinct patient states in specific time periods. Furthermore, different clinical variables are sparsely connected through associated organs, aiding doctors in making comprehensive prediction.

2 Related Work

2.1 Modeling Irregular Clinical Time Series

To learn the sparse and irregular clinical time series, a variety of methods have been developed [7]. Most of these methods are built on recurrent neural networks (RNNs) and typically require complete input data, necessitating modifications to handle missing values. Therefore, various imputation techniques have been proposed to transform sparse irregular data into dense and regular sequences [8, 10]. Another research avenue employs differential equations to represent underlying continuous processes associated with irregularly sampled data [2, 11, 20]. However, these methods are limited in computation efficiency since they require the numerous iterations for differential equation solving. Furthermore, Raindrop [33] approaches the problem by transforming irregular time series into sensor graphs. And graph neural networks are utilized to capture the correlation among different variables. Recently, Transformers have shown the promising effectiveness in capturing long-range dependencies for sequential data, making them an attractive option for time series modeling.

2.2 Transformer for Clinical Time Series

While Transformer-based methods [4, 18, 22] have shown effectiveness with regular temporal data, adapting them for irregular clinical time series presents challenges. Recent methods tailored for clinical time series emphasize custom self-attention mechanisms to capture temporal dependencies and efficient tokenization to understand inter-variable relationships. In clinical time series analysis, it is essential to capture both time and variable dependencies. To tackle the above challenges, [28] takes a different approach by embedding tokens from sequences of (time, variate, value) triplets, achieving global representation through a Transformer. Hi-BEHRT handles the longer sequences of individual observations based on loys a hierarchical Transformer architecture. mTAN [24] proposes an advanced attention-based interpolation method to convert irregular time series into regular continuous-time embeddings. UTDE [34] combines the mTAN embeddings with traditional imputation technique based on the learnable gates, which leverages their strengths in addressing intricate temporal dependency. ContiFormer [3] extends continuous dynamics in ODE to the self-attention mechanism, which adapts the Transformer to the continuous-time domain. Warpformer utilizes the time warping technique to adaptively unify irregular time series into the given scale. Meanwhile, a doubly self-attention module in Warpformer is proposed for representation learning across both time and variable dimensions. Finally, ViTST [13] adapts powerful vision Transformers by converting irregular time series into line graph images.

Though the above methods have been effective at capturing temporal dependency, all of them focus on global dependency, which may inevitably lose fine-grained temporal variations. In addition, the widely utilized channel-mix or channel-independent (accompanied by feature attention) embeddings are limited in high learning complexity in terms of the intrinsic clinical variable correlation.

3 Preliminaries

Let $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{T}_i, y_i)\}_{i=1}^N$ denote a dataset with N patients. Each patient contains a clinical time series $(\mathbf{X}_i, \mathbf{T}_i)$ and a binary prediction label y_i (e.g. discharge or death for modality prediction). The $(\mathbf{X}_i, \mathbf{T}_i)$ represents a series of records $\mathbf{X}_i = [\mathbf{x}_{i,1}, \mathbf{x}_{i,2}, \dots, \mathbf{x}_{i,|\mathbf{T}_i|}]$ and corresponding timestamps $\mathbf{T}_i = [t_{i,1}, t_{i,2}, \dots, t_{i,|\mathbf{T}_i|}]$ at all visits. Each $\mathbf{x}_{i,j}$ includes the values of certain variates in all K clinical variates (lab parameters, physiological signals) at the j -th visit since only partial variates are examined at each visit. For the k -th clinical variate, its series and corresponding timestamp list are denoted as $\mathbf{x}_i^k = [x_{i,1}^k, x_{i,2}^k, \dots, x_{i,|\mathbf{t}^k|}^k]$ and $\mathbf{t}_i^k = [t_{i,1}^k, t_{i,2}^k, \dots, t_{i,|\mathbf{t}^k|}^k]$. In the following part, we drop the patient index i for simplicity.

4 Proposed Method

The architecture of the proposed **Multi-Scale Value-Density Transformer** with **Medical Semantic Guidance (MVMformer)** is shown in Fig. 2. In a nutshell, MVMformer learns clinical time series from segment view based on a stack of Multi-Scale Value-Density Attention (MS-VDAtn) modules while connecting diverse clinical variables based on a Hierarchical Medical Semantic Graph (HMS Graph). Specifically, the MS-VDAtn is introduced to capture both value trend and temporal density for multi-length segments, which considers the irregularity in clinical data. In the top layer, MVMformer establishes hierarchical connections between clinical variables based on their associations with organs through the HMS Graph. In addition, the proposed MVMformer can directly deal with clinical time series without imputation, which avoids unreliable values.

4.1 Time Series Segmentation

Given a fixed length L , we divide the clinical time series into several segments, with each observed time stamp $t_j^k \in \mathbf{t}^k$ serving as the center of a segment. This segmentation process is applied individually to each k -th clinical variable. In addition, the circular padding is applied for the border of each series. Each segment \mathbf{S}_j^k centered at $c = t_j^k$ can be defined as follows:

$$\mathbf{S}_j^k = \left\{ (t, x) \mid c = t_j^k, c - \frac{L}{2} \leq t \leq c + \frac{L}{2}, t \in \mathbf{t}^k \right\} \quad (1)$$

where t^k, x^k are the observed time points and values in $[t_j^k - \frac{L}{2}, t_j^k + \frac{L}{2}]$. We drop the variable index k and timestamp index j for brevity the following section.

4.2 Multi-scale Value-Density Attention

Effective segment modeling in clinical time series needs to capture dependency in both value and time dimension, while accommodating varied segment lengths. In this section, we introduce the proposed Multi-scale Value-Density Attention (MS-VDAtn) module to address these requirements. MS-VDAtn is built upon the kernelized attention [14], which integrates the kernel – a powerful tool for capturing local dependency - into attention mechanism. Unlike kernelized attention focuses on reducing computational complexity, our MS-VDAtn aims to improve the segment representation learning. Specifically, we first enhance kernelized attention in multi-scale modeling, which handles multi-length segments in a single module. Furthermore, we design a value and density units within the MS-VDAtn module, which are tailored to capture both the value trend and temporal density within segments.

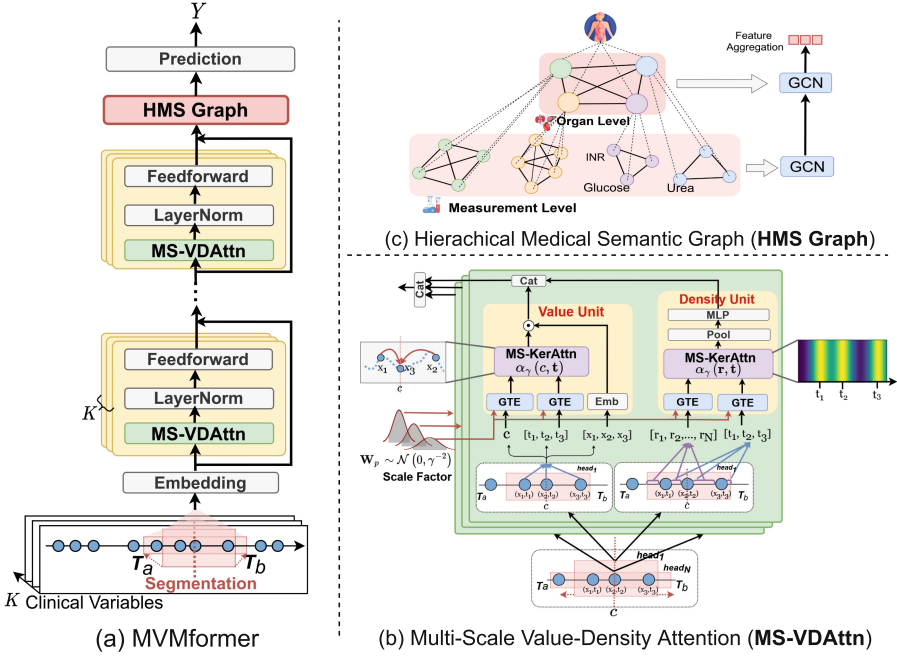


Fig. 2. (a) The overview of the proposed MVMformer model. which consists of: (b) Multi-Scale Value-Density Attention (MS-VDAtn) to model multi-length segments in both value trend and temporal density. (c) Hierarchical Medical Semantic Graph (HMS Graph) to connect the groups of clinical variables with associated organs.

Multi-scale Kernelized Attention. It enhances the kernelized attention in multi-scale modeling, which handles multi-length segments simultaneously. The kernelized attention can be viewed as the kernel approximation in the attention. We focus on the widely utilized γ -scaled Gaussian kernel $\kappa_\gamma(t, t')$, Its kernelized attention can be formulated as:

$$\kappa_\gamma(t, t') \approx \alpha_\gamma(t, t') = \exp(\mathbf{W}_Q \mathbf{p}_t \cdot \mathbf{W}_K \mathbf{p}_{t'}) \tag{2}$$

$$\mathbf{p}_t = \frac{1}{\sqrt{D}} [\cos(w_1 t + b_1), \cos(w_2 t + b_2), \dots, \cos(w_D t + b_D)] \tag{3}$$

where $w_i \sim \mathcal{N}(0, \gamma^{-2})$, $b_i \sim U(0, 2\pi)$. \mathbf{p}_t is time embedding $\Phi_\gamma(\cdot)$ derived from a set of Fourier features, which plays a crucial role in determining the kernel structure and kernel scale γ .

To control the scale of $\alpha_\gamma(t, t')$ while making it adapt to specific data characteristics, we enhance the \mathbf{p}_t to a Gaussian Time Embedding (GTE) $\Phi_{\gamma, \theta}(t)$:

$$\mathbf{p}_{\gamma, \theta}(t) = \frac{1}{\sqrt{D}} [\cos \mathbf{W}_{p\theta} t \parallel \sin \mathbf{W}_{p\theta} t], \mathbf{W}_{p\theta} \sim \mathcal{N}(0, \gamma^{-2}) \tag{4}$$

$$\Phi_{\gamma, \theta}(t) = \text{GeLU}(\mathbf{W}_\theta \mathbf{p}_{\gamma, \theta}(t) + \mathbf{b}_\theta) \tag{5}$$

where the \mathbf{W}_p is adaptively initialized from a normal distribution to ensure approximation with a given scale γ . Moreover, we make \mathbf{W}_p , \mathbf{W}_θ , \mathbf{b}_θ learnable to adapt to specific data characteristics.

Based on the $\Phi_{\gamma,\theta}(t)$, we enhance the multi-scale modeling ability of kernelized attention by incorporating diverse scales to handle various segment lengths across various heads. Different scale factors $\gamma = [\gamma_1, \gamma_2, \dots, \gamma_h]$ are leveraged in h heads. Meanwhile, we conduct segmentations h times using different lengths $\mathbf{L} = [L_1, L_2, \dots, L_h]$ in the MS-VDAtn. The multi-scale kernelized attention handles varied-length segments (centered at the same point) across different heads. This facilitates the identification of dominant segments with dynamic lengths.

Value Unit. it captures the value dependency between the center point and other time points within segment via the $\alpha_\gamma(\cdot)$, as inspired by the continuous kernel convolution. For each segment \mathbf{S}_j^k , the kernelized attention scores are computed by taking its center point c as query and all observed time stamps $\mathbf{t}_S \in \mathbf{S}$ as keys:

$$\mathbf{A}_{temp} = \frac{\alpha_\gamma(c, \mathbf{t}_S)}{\sum_{t \in \mathbf{t}_S} \alpha_\gamma(c, t)} \in \mathbb{R}^{|\mathbf{t}_S|} \quad (6)$$

Then all observed values $\mathbf{x}_S \in \mathbf{S}$ act as the keys. The value embedding vector \mathbf{h}_S^{temp} is obtained by aggregating all observation information within segment:

$$\mathbf{h}_S^{temp} = \mathbf{A}_{temp}(\mathbf{W}_V \mathbf{x}_S) \in \mathbb{R}^d \quad (7)$$

The value unit deals with the observed time points, with α_{temp} normalized to 1 regardless of the number of time points. Therefore, it is necessary to design α to reflect the discrepant intensity among segments.

Density Unit. It extends the kernel density visualization [1] to encode the temporal density by learning the temporal distribution of kernelized attention scores.

To obtain the continuous distribution of kernelized attention scores over time, we first discretized a set of regular reference time stamps $\mathbf{r} = [r_1, r_2 \dots r_M]$ within each segment. Then we compute the kernelized attention matrix \mathbf{A}_{spa} between the observed time embeddings $\Psi(\mathbf{t}_S)$ and reference time embeddings $\Psi(\mathbf{r})$.

$$\mathbf{A}_{int} = \frac{\alpha_\gamma(\mathbf{t}_S, \mathbf{r})}{\sum_{r \in \mathbf{r}} \alpha_\gamma(\mathbf{t}_S, r)} \in \mathbb{R}^{|\mathbf{t}_S| \times |\mathbf{r}|} \quad (8)$$

where \mathbf{A}_{spa} indicates the density between observed points and reference points. We aggregate the \mathbf{A}_{spa} along each reference point:

$$\mathbf{a}_{int} = \text{pool}(\mathbf{A}_{int}) \in \mathbb{R}^{|\mathbf{r}|} \quad (9)$$

where the density attention vector \mathbf{a}_{int} contains the diverse density magnitude across reference points. The \mathbf{a}_{int} is encoded into a hidden vector, which represents the learned temporal density pattern within each segment:

$$\mathbf{H}_{int} = \mathbf{A}_{int} \mathbf{W}_p \quad (10)$$

where the $\mathbf{H}_{int} \in \mathbb{R}^{|\mathbf{t}_s| \times d}$, $\mathbf{W}_p \in \mathbb{R}^{|\mathbf{r}| \times d}$. Then we aggregate \mathbf{H}_{int} to obtain the intensity embedding for each segment:

$$\mathbf{h}_S^{int} = \text{MLP}(\mathbf{a}_{int}) \in \mathbb{R}^d \quad (11)$$

We concatenate the representations from the value unit and intensity unit as the final output at each segment \mathbf{S} :

$$\mathbf{h}_S = [\mathbf{h}_S^{temp}, \mathbf{h}_S^{int}] \quad (12)$$

4.3 Hierarchical Medical Semantic Graph

In this section, we establish connections between various clinical variables by identifying their correlations through a hierarchical medical semantic graph (HMS Graph). The HMS Graph mimics the hierarchical diagnosis process that starts with specific examinations, then moves toward corresponding organ functions, and finally reaches a final diagnosis. As illustrated in Fig. 2(c), we construct a two-level medical semantic graph:

Measurement-Level Graph Construction. We categorize all clinical variables into distinct groups based on public medical knowledge [17], with each group associated with a particular organ system. Subsequently, we transform this variates group structure into a correlation graph.

$\mathcal{G}_o(\mathcal{V}_o, \mathcal{E}_o)$ is a undirected graph, where nodes \mathcal{V} denote variates groups associated with organ type $o \in \mathcal{O}$ and edges \mathcal{E} describe dependencies between variates. Each node vector \mathbf{v}_k in \mathcal{V}_o is encoded that summarizes all segment embeddings along each variable dimension:

$$\mathbf{v}_k = \text{pool} \left(\left[\mathbf{h}_{\mathbf{S}_1^k}, \dots, \mathbf{h}_{\mathbf{S}_{|\mathbf{t}_k|}^k} \right] \right), 1 \leq k \leq K \quad (13)$$

The edge weights are randomly initialized as fully-connected graphs and learnable with prediction loss. Besides, we build a virtual node \mathbf{v}_o to aggregate variable group information, which represents the organ state.

Organ-Level Graph Construction. At the top level, we construct the organ-level graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ to provide a holistic view of the status of different organs: Each graph node $\mathbf{v}_o \in \mathcal{V}$ denotes the organ states inferred from the variate group. Edge $e_{o,*} \in \mathcal{E}$ denotes the interactions between organs.

Hierarchical Graph-based Interaction. We learn the HMS Graph based on two layers of GCNs networks, with information passing from measurement-level and organ-level graphs. In the first GCN layer, feature interactions occur between variate nodes within the same group, capturing intra-group variable correlations:

$$[\mathbf{v}_1, \dots, \mathbf{v}_{|\mathcal{V}_o|}, \mathbf{v}_o] = \text{GCN}_1(\mathcal{G}_o(\mathcal{V}_o, \mathcal{E}_o)), o \in \mathcal{O} \quad (14)$$

After $\text{GCN}_1(\cdot)$, the virtual (organ) node combines information from all corresponding clinical variables. In the second layer of GCN, all the virtual nodes exchange information with each other to analyze organ states and their interactions.

$$\left[\mathbf{v}_1^*, \dots, \mathbf{v}_o^*, \dots, \mathbf{v}_{|\mathcal{O}|}^* \right] = \text{GCN}_2(\mathcal{G}(\mathcal{V}, \mathcal{E})) \quad (15)$$

Finally, the compact and personalized prediction is obtained by aggregating information from the representations:

$$y = \text{Pred} \left(\text{pool} \left(\left[\mathbf{v}_1^*, \dots, \mathbf{v}_o^*, \dots, \mathbf{v}_{|\mathcal{O}|}^* \right] \right) \right) \quad (16)$$

5 Experiments

We evaluate our method in the disease prediction task using three real-world medical datasets.

5.1 Experimental Setup

Datasets. We utilize three real-world medical datasets. The statistical summary of these datasets is shown in the Table 1.

Physionet 2012 (P12)¹ comprises 11,988 clinical time series of patients, including 35 physiological and laboratory parameters, alongside a binary label denoting patient survival during hospitalization.

Physionet 2019 (P19)² consists of 37,320 clinical time series of patients, including 34 physiological and laboratory parameters, along with a binary label indicating the presence of sepsis.

MIMIC-III (MIMIC)³ includes over 58,000 hospital admissions from Beth Israel Deaconess Medical Center, covering the years 2001 to 2012. We focus on the first 48 h of records, which encompass 28 physiological and laboratory parameters [5]. Our experiments involve conducting a mortality prediction task.

Table 1. Statistics of P12, P19 and MIMIC datasets.

	Patients	Average visits (per patient)	Clinical variates	Missing rate	Positive rate
P12	11,988	45.6	35	80.4%	10.1%
P19	37,320	44.54	34	79.8%	2.2%
MIMIC	28,951	42.3	28	82.9%	11.8%

Baselines. We compare our method with several types of baselines tailored to clinical time series prediction as follows:

¹ <https://physionet.org/content/challenge-2012/1.0.0/>.

² <https://physionet.org/content/challenge-2019/1.0.0/>.

³ <https://physionet.org/content/mimiciii/1.4/>.

- **RNN**
 - **DATA-GRU** [27] introduces irregularity-sensitive updating mechanisms into recurrent neural network
- **GNN**
 - **Raindrop** [33] transforms the clinical time series into a set of variable tuples and models their correlations by graph neural networks.
- **Transformer**
 - **ViTST** [13] converts irregular time series into line images and utilizes advanced vision transformers to capture representations of the time series.
 - **DuETT** [9] transforms sparse time series into a regular sequence and incorporates both temporal and feature attentions to facilitate robust representation extraction.
 - **Warpformer** [32] combines self-attention module with a warping module to dynamically unify irregular time series within a specified scale.

Implementation. We randomly split each dataset into three subsets: a training set, a validation set, and a testing set, adhering to an 8:1:1 ratio. All experiments were performed on a server with an Intel Core 2.80 GHz processor and an Nvidia Tesla V100 GPU, employing 5-fold cross-validation. The batch size was set to 16, and the proposed MVMformer was trained for 100 epochs using the Adam optimizer with a learning rate of 0.0005. We configured the number of segment layers to 3, with the $h = 8$, $d_t = 64$, and $d_x = 32$ in VI-KerAttn for each layer. The bandwidth parameter γ_h was set to $\frac{1}{L_h}$, scaling with the input segment length. To enhance multi-scale modeling capabilities, we utilized progressive segment lengths ((5, 9, 13, 17), (13, 17, 21, 25), (21, 25, 29, 33)) across the three layers, with each segment length processed by 2 heads. This hyperparameter setup exhibited strong generalization performance across all three datasets. The evaluation metrics include the Area Under the ROC Curve (AUROC) and the Area Under the Precision-Recall Curve (AUPRC).

Table 2. Performance comparison for prediction accuracy in three real-world medical datasets

Type	Methods	P12		P19		MIMIC	
		AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
RNN	DATA-GRU [25]	84.2±0.5	45.2±0.4	81.9±0.5	36.9±0.6	85.1±0.7	49.4±0.5
GNN	Raindrop [33]	81.9±0.8	43.1±0.9	83.2±0.7	38.3±0.8	83.6±0.5	47.7±0.6
Transformer	ViTST [13]	84.9±0.6	46.8±0.5	82.7±0.5	37.2±0.6	85.9±0.6	50.5±0.7
	DuETT [9]	85.4±0.7	47.5±0.7	<u>84.1 ± 0.6</u>	40.3±0.5	87.1±0.5	52.1±0.6
	Warpformer [32]	<u>86.1 ± 0.6</u>	<u>48.2 ± 0.5</u>	82.7±0.8	<u>41.5 ± 0.7</u>	<u>87.8 ± 0.4</u>	<u>53.8 ± 0.4</u>
	MVMformer	87.5±0.5	50.4±0.4	85.9±0.5	43.9±0.6	89.5±0.5	56.8±0.5

5.2 Prediction Accuracy

Table 2 shows the experimental outcomes of MVMformer alongside all baseline methods regarding AUROC and AUPRC across three real-world datasets.

Among the baselines, Warpformer and DuETT perform the best accuracy, highlighting the effectiveness of Transformer architecture for time series learning. Despite DuETT utilizing variate-independent tokenization and attention mechanisms across both time and variate dimensions, it does not surpass Warpformer. This is primarily due to the extensive hypothesis space and increased learning complexity when capturing all pair-wise variate correlations through self-attention. In contrast, MVMformer consistently outperforms both Warpformer and DuETT across all datasets, with manageable variance. This advantage stems from MVMformer’s ability to integrate informative local dynamics, which are often overlooked by other Transformer-based methods. Additionally, MVMformer constructs a medical semantic graph by leveraging medical knowledge, which simplifies the learning of variate correlations into several sub-problems focused on specific variate groups. This approach reduces the hypothesis space and lowers learning complexity, resulting in enhanced performance. This is particularly evident in datasets with fewer samples and a higher number of variates (e.g., P12), where the high sample complexity required for capturing complete feature correlations is not met. MVMformer effectively addresses this issue, achieving superior results. By comparing results between the P12 and MIMIC, we can find: in the P12, which has fewer samples and a higher missing rate, the performance improvements are 4.23% for AUPRC and 1.5% for AUROC compared to the best baselines. These gains significantly surpass those seen in the MIMIC dataset, which shows improvements of 2.67% for AUPRC and 1.25% for AUROC.

Table 3. Ablation study for the proposed MS-VDKerAttn

ID	Ablations				P12		P19		MIMIC	
	KerAttn	Multi-Scale	Value	Density	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
①					83.1±0.8	46.8±0.5	82.1±0.5	40.2±0.7	85.7±0.6	51.5±0.7
②	✓				84.2±0.9	47.5±0.7	82.9±0.4	41.7±0.6	86.2±0.7	52.1±0.5
③	✓	✓			85.3±0.6	48.2±0.6	84.1±0.5	42.5±0.6	87.1±0.6	54.3±0.5
④	✓	✓	✓		86.1±0.7	49.3±0.4	85.1±0.7	43.1±0.8	87.8±0.4	55.2±0.6
⑤	✓	✓		✓	86.7±0.7	49.7±0.4	85.4±0.7	43.1±0.8	88.5±0.4	55.9±0.6
⑥			✓	✓	84.9±0.6	47.9±0.7	84.0±0.6	42.1±0.7	86.8±0.7	53.4±0.8
Ours	✓	✓	✓	✓	87.5±0.5	50.4±0.4	85.9±0.5	43.9±0.6	89.5±0.5	56.8±0.5

5.3 Ablation Study

We conducted ablation studies on MVMformer to validate the significance of critical designs: (1) Multi-Scale Value-Density Attention (MS-VDKerAttn) (2) Hierarchical Medical Semantic Graph (HMS Graph).

MS-VDKerAttn. We construct 6 variants for MS-VDKerAttn: (1) For ① and ⑥, we replace multi-scale kernelized attention with vanilla self-attention. (2) For ②, we remove multi-scale modeling ability in MS-VDKerAttn. (4) For ③, ④, and ⑤, we separately remove the value or density unit in the MS-VDKerAttn.

Table 3 illustrates the comparison results. We can discover that ② outperforms ① in all datasets, which indicates that the kernelized attention is more appropriate for segment modeling compared to vanilla attention. Meanwhile, ③ outperforms ②, demonstrating that incorporating multi-scale modeling into the kernelized attention enhances the generalization ability for dynamic segment lengths. Moreover, we observe that ④ and ⑤ outperform ③ with ⑤ marginally outperforming ①. This highlights the universal effectiveness of both the value and density units in both kernelized and non-kernelized attentions. Notably, our proposed method performs better than ⑤. This indicates that our effective multi-scale kernelized attention further improves the performance of the value and density units.

Table 4. Ablation study for the proposed HMS Graph

ID	Ablations		P12		P19		MIMIC	
	Semantics	Hierarchical	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
①			83.7±0.7	46.7±0.8	82.5±0.6	40.2±0.7	85.9±0.7	53.6±0.9
②	✓		86.8±0.6	49.2±0.5	85.2±0.5	42.8±0.6	87.8±0.6	55.9±0.5
③		✓	85.2±0.6	47.8±0.7	84.1±0.5	41.3±0.5	86.4±0.6	54.4±0.7
Ours	✓	✓	87.5±0.5	50.4±0.4	85.9±0.5	43.9±0.6	89.5±0.5	56.8±0.5

HMS Graph. Table 4 includes the results of two variants: (1) ① replaces the HMS Graph with a single-level and full-connected graph, where both node embeddings and correlation weights are randomly initialized and learnable. (2) ② constructs a hierarchical graph without medical semantic affinity, where variable groups are built by clustering. (3) ③ replaces the HMS Graph with a single-level and full-connected graph while preserving medical semantics by assigning a correlation weight of 1 for similar clinical variable.

From Table 4, we can observe that ② outperforms ① since our hierarchical graph indirectly regularizes the sparse similarity between clinical variables and thus alleviates learning complexity. Moreover, ③ outperforms ①, demonstrating that medical semantic affinity can effectively connect diverse clinical variables. Notably, our MVMformer outperforms ② and ③, indicating that the medical semantic affinity into the hierarchical graph structure is compatible with the medical semantic affinity of clinical variables and adaptively combining them can further improve model performance.

5.4 Analysis of HMS Graph in Missing Variable Settings

Our MVMformer can implicitly address missing variables by leveraging the learned dependencies among clinical variables in the HMS Graph. To evaluate this capability, we assess whether MVMformer can perform well when a subset of variables is entirely absent. This scenario is particularly relevant in cases where certain variables may not be available at specific institutions. In this setting, we keep the training samples unchanged. For both validation and test sets, we randomly select a portion of the variables and conceal all their observations. The excluded variables remain the same across samples and models.

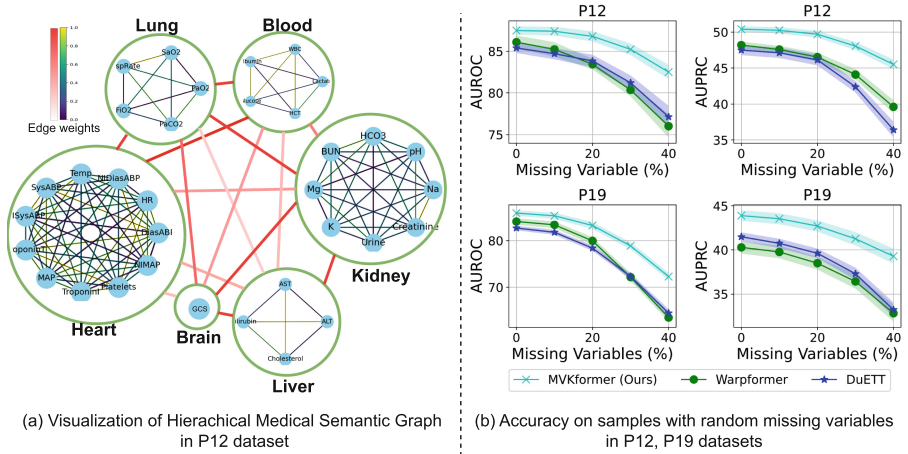


Fig. 3. Analysis of HMS Graph in Missing Variable Settings.

HMS Graph Visualization. As illustrated in Fig. 3(a), we visualize the learned HMS Graph. Distinct patterns can be observed in the two-level learned dependency graphs, demonstrating that MVMformer can adaptively learn both inter-variable and inter-organ graph structures that are sensitive to the prediction task. In addition to the visualization, we can derive several concrete insights. For example, the function of the “Heart” system is notably affected by both the “Lung” and “Blood” systems. Furthermore, within the lung system, there is a strong dependency among “RespRate,” “PaO2,” and “SaO2.”

Accuracy in Missing Variable Settings. We report results of both P12 and P19 datasets in Fig. 3(b). We find that MVMformer achieves the highest performance in both AUROC and AUPRC when the percentage of missing variables ranges from 10% to 40%. As the amount of missing data increases, MVMformer demonstrates significant performance enhancements with consistent predictions. It outperforms baseline models by as much as 24.9% in AUROC and 29.3% in AUPRC. This is primarily attributed to the capability of the proposed HMS Graph to group variables with similar medical semantics, allowing the information from partially missing variables to be supplemented by their dependencies on others.

6 Conclusion

In this paper, we propose a MVMformer to perform disease prediction based on clinical time series. In contrast with existing methods, MVMformer takes irregularity-aware segment-wise modeling for clinical time series while considering the medical semantic affinities among clinical variables, consisting of two

key modules. A Multi-scale Value-Density Attention is first introduced to capture both value trend and temporal density within multi-length segments. Then a Hierarchical Medical Semantic Graph is constructed to learn the group similarity of clinical variables with associated organs, leveraging public medical knowledge.

Acknowledgments. This work was supported by Hong Kong Research Grants Council General Research Fund under Grant RGC/HKBU12200122.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Chan, T.N., Hou, U.L., Choi, B., Xu, J., Cheng, R.: Kernel density visualization for big geospatial data: algorithms and applications. In: 2023 24th IEEE International Conference on Mobile Data Management (MDM), pp. 231–234 (2023). <https://doi.org/10.1109/MDM58254.2023.00046>
2. Chen, Y., et al.: Provably convergent schrödinger bridge with applications to probabilistic time series imputation. In: International Conference on Machine Learning, pp. 4485–4513. PMLR (2023)
3. Chen, Y., Ren, K., Wang, Y., Fang, Y., Sun, W., Li, D.: Contiformer: continuous-time transformer for irregular time series modeling. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
4. Gruver, N., Finzi, M., Qiu, S., Wilson, A.G.: Large language models are zero-shot time series forecasters. In: Advances in Neural Information Processing Systems, vol. 36 (2024)
5. Harutyunyan, H., Khachatrian, H., Kale, D.C., Ver Steeg, G., Galstyan, A.: Multitask learning and benchmarking with clinical time series data. *Sci. data* **6**(1), 96 (2019)
6. Jia, F., Wang, K., Zheng, Y., Cao, D., Liu, Y.: Gpt4mts: prompt-based large language model for multimodal time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 23343–23351 (2024)
7. Jin, M., et al.: A survey on graph neural networks for time series: forecasting, classification, imputation, and anomaly detection. *IEEE Trans. Pattern Anal. Mach. Intell.* (2024)
8. Kim, S., Kim, H., Yun, E., Lee, H., Lee, J., Lee, J.: Probabilistic imputation for time-series classification with missing data. In: International Conference on Machine Learning, pp. 16654–16667. PMLR (2023)
9. Labach, A., et al.: Duett: dual event time transformer for electronic health records. In: Machine Learning for Health (ML4H) (2023)
10. Li, D., Lyons, P., Klaus, J., Gage, B., Kollef, M., Lu, C.: Integrating static and time-series data in deep recurrent models for oncology early warning systems. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 913–936 (2021)
11. Li, J., Zhu, Z., et al.: Neural lad: a neural latent dynamics framework for times series modeling. In: Advances in Neural Information Processing Systems, vol. 36 (2024)

12. Li, T., Liu, Z., Shen, Y., Wang, X., Chen, H., Huang, S.: Master: market-guided stock transformer for stock price forecasting. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 162–170 (2024)
13. Li, Z., Li, S., Yan, X.: Time series as images: vision transformer for irregularly sampled time series. In: *Advances in Neural Information Processing Systems*, vol. 36 (2024)
14. Luo, S., et al.: Stable, fast and accurate: kernelized attention with relative positional encoding. In: *Advances in Neural Information Processing Systems*, vol. 34, pp. 22795–22807 (2021)
15. Lyu, F., Ye, M., Carlsen, J.F., Erleben, K., Darkner, S., Yuen, P.C.: Pseudo-label guided image synthesis for semi-supervised covid-19 pneumonia infection segmentation. *IEEE Trans. Med. Imaging* **42**(3), 797–809 (2022)
16. Lyu, F., Ye, M., Ma, A.J., Yip, T.C.F., Wong, G.L.H., Yuen, P.C.: Learning from synthetic ct images via test-time training for liver tumor segmentation. *IEEE Trans. Med. Imaging* **41**(9), 2510–2520 (2022)
17. McClatchey, K.D.: *Clinical laboratory medicine*. Lippincott Williams & Wilkins (2002)
18. Ni, Z., Yu, H., Liu, S., Li, J., Lin, W.: Basisformer: attention-based time series forecasting with learnable and interpretable basis. In: *Advances in Neural Information Processing Systems*, vol. 36 (2024)
19. Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: long-term forecasting with transformers. In: *International Conference on Learning Representation* (2022)
20. Oskarsson, J., Sidén, P., Lindsten, F.: Temporal graph neural networks for irregular data. In: *International Conference on Artificial Intelligence and Statistics*, pp. 4515–4531 (2023)
21. Pan, F., Yin, C., Liu, S.Q., Huang, T., Bian, Z., Yuen, P.C.: Bindingsitedti: differential-scale binding site modelling for drug–target interaction prediction. *Bioinformatics* **40**(5), btac308 (2024)
22. Ren, H., Wang, J., Zhao, W.X., Wu, N.: Rapt: pre-training of time-aware transformer for learning robust healthcare representation. In: *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 3503–3511 (2021)
23. Reyna, M.A., et al.: Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. In: *2019 Computing in Cardiology (CinC)*. IEEE (2019)
24. Shukla, S.N., Marlin, B.: Multi-time attention networks for irregularly sampled time series. In: *International Conference on Learning Representation* (2021)
25. Tan, Q., et al.: UA-CRNN: uncertainty-aware convolutional recurrent neural network for mortality risk prediction. In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 109–118 (2019)
26. Tan, Q., Ye, M., Wong, G.L.H., Yuen, P.C.: Cooperative joint attentive network for patient outcome prediction on irregular multi-rate multivariate health data. In: *International Joint Conference on Artificial Intelligence* pp. 1586–1592 (2021)
27. Tan, Q., et al.: Data-GRU: dual-attention time-aware gated recurrent unit for irregular multivariate time series. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, pp. 930–937 (2020)
28. Tipirneni, S., Reddy, C.K.: Self-supervised transformer for sparse and irregularly sampled multivariate clinical time-series. *ACM Trans. Knowl. Disc. Data (TKDD)* **16**(6), 1–17 (2022)

29. Xu, J., Lyu, F., Yuen, P.C.: Density-aware temporal attentive step-wise diffusion model for medical time series imputation. In: Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, pp. 2836–2845 (2023)
30. Xu, J., Zhu, Y., Lyu, F., Wong, G.L.H., Yuen, P.C.: Temporal neighboring multi-modal transformer with missingness-aware prompt for hepatocellular carcinoma prediction. In: Linguraru, M.G., et al. (eds.) MICCAI 2024, pp. 79–88. Springer, Heidelberg (2024). https://doi.org/10.1007/978-3-031-72378-0_8
31. Yin, C., et al.: Xfibrosis: explicit vessel-fiber modeling for fibrosis staging from liver pathology images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 11282–11291 (2024)
32. Zhang, J., Zheng, S., Cao, W., Bian, J., Li, J.: Warpformer: a multi-scale modeling approach for irregular clinical time series. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (2023)
33. Zhang, X., Zeman, M., Tsiligkaridis, T., Zitnik, M.: Graph-guided network for irregularly sampled multivariate time series. In: International Conference on Learning Representation (2021)
34. Zhang, X., Li, S., Chen, Z., Yan, X., Petzold, L.R.: Improving medical predictions by irregular multimodal electronic health records modeling. In: International Conference on Machine Learning, pp. 41300–41313 (2023)
35. Zhu, Y., Xu, J., Lyu, F., Yuen, P.C.: Symptom disentanglement in chest x-ray images for fine-grained progression learning. In: Linguraru, M.G., et al. (eds.) MICCAI 2024, pp. 598–607. Springer, Heidelberg (2024). https://doi.org/10.1007/978-3-031-72378-0_56



Variational Autoencoder Learns Better Feature Representations for EEG-Based Obesity Classification

Yuan Yue¹(✉), Dirk De Ridder², Patrick Manning³, and Jeremiah D. Deng¹

¹ School of Computing, University of Otago, Dunedin, New Zealand
yueyu445@student.otago.ac.nz

² Department of Surgical Science, University of Otago, Dunedin, New Zealand

³ Department of Medicine, University of Otago, Dunedin, New Zealand

Abstract. Obesity is a common issue in modern societies today that can lead to significantly reduced quality of life. Existing research on investigating obesity-related neurological characteristics is limited to traditional approaches such as significance testing and regression. These approaches may require certain neurological assumptions to be made and may struggle to handle the complexity and non-linear relationships within the high-dimensional electroencephalography (EEG) data. In this study, we propose a deep learning-based approach for extracting features from resting-state EEG signals to classify obesity-related brain activity. Specifically, we employ a Variational Autoencoder (VAE) to learn robust feature representations from EEG data, followed by classification using a 1-D convolutional neural network (CNN). By comparing our approach with benchmark models, we demonstrate the efficiency of VAE in feature extraction, evidenced by significantly improved classification accuracies, enhanced visualizations, and reduced impurity measures in the learned feature representations.

Keywords: Deep learning · EEG classification · Variational Autoencoder

1 Introduction

Obesity is a global health issue linked to dysfunction in multiple body systems, including the heart, liver, kidneys, joints, and reproductive system [1–4]. It also contributes to the onset of various diseases, such as type 2 diabetes, cardiovascular diseases, and cancers [5]. While much research focuses on the clinical characteristics of obesity, increasing attention is being directed towards its neurological effects [6, 7]. Techniques such as electroencephalography (EEG) and functional magnetic resonance imaging (fMRI), which capture brain structure and activity, have been extensively employed to investigate brain activity patterns related to obesity [8–10].

Several studies have suggested that obesity is linked to cognitive impairment and altered brain network structures [8, 11]. For instance, abnormal connectivity in the somatosensory cortex and insula in obese individuals may impair their ability to predict energy needs, leading to overeating [12]. Additionally, altered hippocampal structures, which are strongly correlated with dementia, have been observed in obese individuals [13].

However, obesity-related brain activities are often researched using traditional statistical approaches such as significance testing and regression, which focus on predefined, isolated brain regions [14, 15]. These approaches have limitations: they may not capture complex interactions between brain regions, overlook the holistic brain function where regions interact dynamically, and introduce bias by focusing on predefined areas based on prior knowledge. Additionally, such approaches may not account for individual variability in brain structure and function. Moreover, brain activity analysis in obesity research often relies on predefined measures like functional connectivity or spectral powers [14, 16]. These measures require prior selection of specific regions or frequency bands, potentially excluding relevant data outside these categories.

These limitations underscore the need for applying machine learning (ML) and deep learning (DL) approaches in obesity-related brain activity research. These approaches can handle the high-dimensional nature of brain signals without significant information loss and automatically detect complex interactions across the entire brain. This enables a more comprehensive analysis of brain patterns, ultimately leading to more effective obesity interventions and treatments.

To the best of our knowledge, no prior research has directly investigated obesity-related brain activities using EEG data through ML or DL approaches. This study aims to address this gap by employing a DL-based approach to learn robust and comprehensive feature representations of brain activity associated with obesity directly from EEG signals.

Specifically, we employ an unsupervised Variational Autoencoder (VAE) to learn high-level, robust feature representations. VAE is chosen in this study due to previous studies have consistently demonstrated its efficiency in robust feature learning compared to other unsupervised feature learning approaches such as autoencoder and Principle Component Analysis [17–19]. These representations are then used as input for a 1-D convolutional neural network (CNN) to perform the obesity classification task. The entire process of our study is demonstrated in Fig. 1. Key novelties of our study include:

- Our study is the first to identify obese brain activities at the EEG signal level using a DL-based framework. It addresses the limitations of traditional approaches by eliminating the need for predefined features and capturing intricate brain signal patterns;
- We demonstrated the effectiveness of using a VAE to learn feature representations for analyzing brain activity associated with obesity.;
- We introduced a quantitative measure based on impurity to evaluate the separability of the feature space, further confirming the superiority of VAE fea-

tures. These enhancements contribute to the interpretability of our proposed model.

We have made the source code for this study available.¹

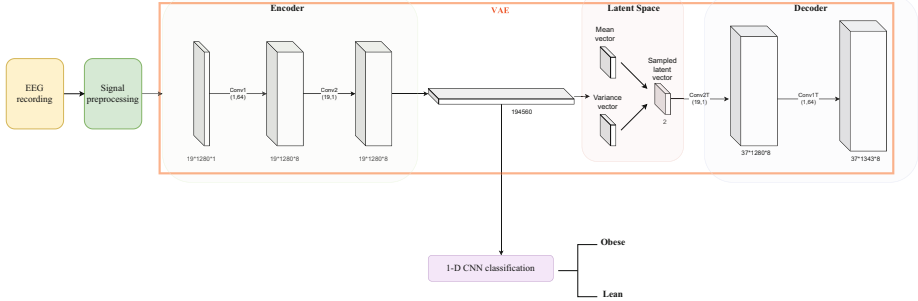


Fig. 1. Overall process of the proposed work.

2 Method

2.1 Data Description and Preprocessing

The EEG datasets used in this study were collected from 30 obese females and 30 lean (i.e., healthy) females who are between 25 to 65 years old. Subjects who have a body mass index (BMI) higher than 30 are defined as obese individuals and those who have a BMI lower than 25 are defined as lean individuals. Given that several studies have reported sex-related differences in obesity and research in the field supports the validity of investigating obesity within a single-sex [20–22], in this study, we limited our research to females only to eliminate potential confounding factors related to sex differences and to gain sex-specific insights.

The international 10–20 sensor layout system, which has 19 main channels, was used for the EEG recording process. For each subject, resting-state EEG data were recorded while fasting and with eyes closed for approximately 5 min.

Regarding the data preprocessing process, the first five seconds of each EEG recording were discarded as they usually contain a high level of noise. The recordings were then resampled to 128 Hz and band-pass filtered between 0.1 Hz and 45 Hz. Next, each recording was segmented into 10-second consecutive epochs, and each epoch would be used as an individual data sample. This resulted in each subject having a total of 26 epochs, meaning each subject’s data consisted of 26 data samples.

The ethical approval for subject recruitment and data collection processes was obtained from the Southern District Health Board Ethics Committee, New Zealand (Ref: 15/STH/68), in accordance with the Declaration of Helsinki.

¹ <https://github.com/2duck1lion/VAE/tree/main>.

2.2 Latent Feature Representation Learning

In this study, we used a VAE model to learn latent feature representations, chosen based on previous studies demonstrating its efficacy in robust feature learning through maintaining a smooth and continuous latent feature space [17–19]. Specifically, VAE operates within a probabilistic framework that facilitates the learning of latent variables capturing the underlying data structure. This capability enables efficient noise reduction and extraction of salient features, thereby enhancing discriminative power in classification tasks.

VAE works by encoding an input data sample into a Gaussian distribution within the latent space, with the decoder subsequently performing the decompression and reconstruction processes using the data points sampled from this space. Additionally, regularization within the latent space is enforced by guiding the encoded latent variables towards this Gaussian distribution. The objective function of a VAE comprises two components: minimizing the reconstruction error to enhance the precision of the encoding-decoding process, and ensuring that the distribution outputted by the encoder closely resembles a Gaussian distribution. The loss function of a VAE can be formally written as:

$$\mathcal{L}_{\text{VAE}} = E(\|\mathbf{x} - \hat{\mathbf{x}}\|^2) + D_{\text{KL}}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})), \quad (1)$$

where \mathbf{z} is the latent variable, $q(\mathbf{z}|\mathbf{x})$ is the latent distribution generated by the encoder, $p(\mathbf{z})$ is the prior distribution in the latent space which follows a Gaussian distribution, and $D_{\text{KL}}(\cdot)$ denotes the Kullback-Leibler divergence.

In this study, we used the encoder output as the learned latent feature representation for classification. This choice is made based on two key considerations: first, the latent representation, being used for reconstructing the input data, is presumed to encapsulate the majority of information from the input; second, leveraging this high-level latent representation helps in filtering out noise unrelated to obesity.

The detailed architecture of the proposed VAE is shown in Fig. 1. This architecture is designed based on the concept of EEGNet [23], which is an extensively used DL model for EEG classification tasks. It mainly consists of a temporal convolution block followed by a depth-wise spatial convolution block. A separable convolution block is then applied to improve the model’s performance. In this study, in the encoding part of the proposed VAE, we first adapted a temporal convolution layer with a kernel size of $1 \times 128/2 = 1 \times 64$, to extract temporal features. We then performed a spatial convolution by using kernels with a size of 19×1 , to extract spatial features. Each convolution layer is followed by a batch normalization layer and a leaky ReLU layer. The decoder is then designed by taking the inverse of the encoder.

2.3 Data Partitioning and Classification

Ten-fold subject-based cross-validation was employed as the data partitioning strategy in this study. Within each fold, 6 subjects (3 lean and 3 obese) were

set aside for testing, while 54 subjects (27 lean and 27 obese) were used for training. Additionally, within each training fold, 6 subjects (3 obese and 3 lean) were further selected for validation. The final testing score (i.e., accuracy) was obtained by averaging the scores across all folds.

For classification, we employed a 1-D CNN, a Support Vector Machine (SVM) with a Radial Basis Function kernel, a K-nearest neighbours (KNN) classifier with $k = 3$, a Random Forest (RF) classifier, and an Adaboost (ADB) classifier to predict the label for each data sample. Hyperparameters were optimized using the validation dataset. The SVM used a gamma parameter set to the inverse of the number of features to control the decision boundary's shape. The RF consisted of 50 estimators with a maximum depth of 10. For ADB, 75 estimators were employed with a learning rate of 0.88. We chose these simple classifiers in this study to emphasize the effectiveness of the learned feature representations.

The architecture of the proposed 1-D CNN is shown in Fig. 2. The network begins with two convolutional layers, employing 8 filters of size 64 in the first layer and 16 filters of size 32 in the second. These are followed by an average pooling layer with a size of 4 and a dropout layer with a rate of 0.25 to minimize the number of parameters and reduce overfitting. The final convolutional layer consists of 32 filters with a size of 16, succeeded by an average pooling layer of size 8 and another dropout layer with a rate of 0.25. After each convolution, batch normalization and activation layers are included to facilitate faster learning and improve convergence through regularization.

For each subject, the final classification label was determined by the majority vote of 26 data samples classified as either obese or lean.

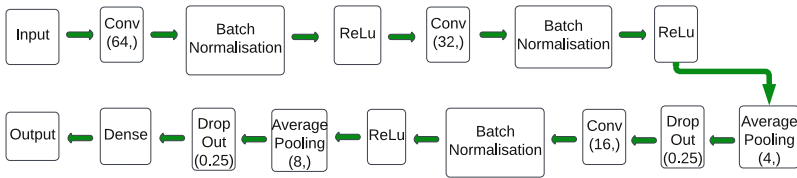


Fig. 2. Architecture of the proposed 1-D CNN used for classification.

2.4 Neural Network Model Training and Optimization

TensorFlow with the Keras API was employed in this study. The VAE was trained on preprocessed EEG data matrices comprising 19 channels and 1280 time points, processed in batches of 26 samples. Training used the Adam optimizer with a learning rate of 0.0005, running for up to 500 epochs with early stopping based on model loss.

The 1-D CNN was optimized using the Adam optimizer with a learning rate of 0.001 and compiled with binary cross-entropy loss. The training was conducted in batches of 200 samples for a maximum of 200 epochs, with early stopping implemented based on validation loss and patience of 20 epochs.

2.5 Impurity-Based Measure for Assessing Feature Discriminability

In this study, we introduced a novel impurity-based measure to assess the discriminant ability of the learned feature representation. Suppose a D -dimension, N -entry feature set \mathbf{X} . Denote the value set of the i -th attribute as X_i , $i = 1, \dots, D$. Using the idea of the decision tree, we seek an optimal threshold τ that splits the values in X_i into two value subsets with minimum impurity:

$$\begin{aligned} X_i^L &= \{x_{ij} | x_{ij} < \tau, j = 1, \dots, N\} \\ X_i^R &= \{x_{ij} | x_{ij} \geq \tau, j = 1, \dots, N\} \end{aligned} \quad (2)$$

As we are dealing with a two-class problem, the impurity can be calculated using the Gini index [24]. Suppose within a subset S , p is the probability of an instance x belonging to Class 1, the Gini index is

$$G(S) = p(1 - p). \quad (3)$$

Hence we define a ‘‘dichotomy impurity’’ (DI) for the i -th attribute based on the minimal weighted average of impurities of the two subsets generated by the best ‘‘cut’’:

$$DI_i = \min_{\tau} \left(\frac{|X_i^L|}{|X_i|} G(X_i^L) + \frac{|X_i^R|}{|X_i|} G(X_i^R) \right), \quad (4)$$

where $|\cdot|$ indicates cardinality. In other words, DI_i indicates the purest dichotomy we can get on attribute i . The overall separability of the feature representation can be roughly indicated by the average DI :

$$DI = \sum_i DI_i / D. \quad (5)$$

The smaller DI is, the better separability we can achieve.

2.6 Benchmark Model Choices and Performance Comparison

As extensively used models for EEG classification tasks, we chose EEGNet, Shallow ConvNet, and Deep ConvNet as the baseline models to compare with our proposed model [23, 25]. For state-of-the-art comparisons, we selected various neural network-based models with diverse architectures that are commonly employed for resting-state EEG classification. These models include VGGNet [3], a CNN-based architecture with deep layers and attention-based residual-inception modules designed to learn complex features; MuLHiTA [26], which focuses on identifying mental stress levels using a multi-branch long short term memory (LSTM) architecture and hierarchical temporal attention mechanisms; AgeNet [27], a deep CNN proposed for age prediction; ParkinsonNet [28], a CNN with LSTM mechanism proposed for Parkinson’s Disease detection; and CognitionNet [29], a multi-head CNN based on attention mechanisms proposed to predict cognitive decline. Given that each of these benchmark models uses an end-to-end classification process that includes distinct approaches for feature extraction and feature embedding, we directly applied them to the preprocessed EEG signals.

3 Results and Discussion

3.1 Classification Performance Comparison

The test scores (i.e., accuracy) obtained using the 1-D CNN, SVM, KNN, RF, and ADB are shown in Table 1. Among these, the 1-D CNN achieved the highest classification scores, with 0.951 for subject-level classification and 0.937 for epoch-level classification.

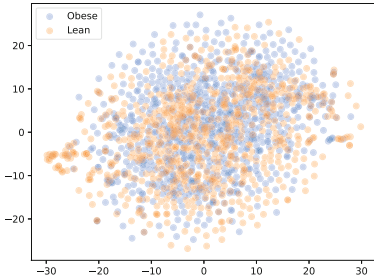
Further comparison of test accuracies between the benchmark models and our proposed model is presented in Table 2. Our proposed model demonstrated its efficiency by significantly outperforming the benchmark models on both subject-level and sample-level classifications.

Table 1. Test scores (mean accuracy \pm standard deviation) obtained using 1-D CNN, SVM, KNN, RF, and ADB as classifiers.

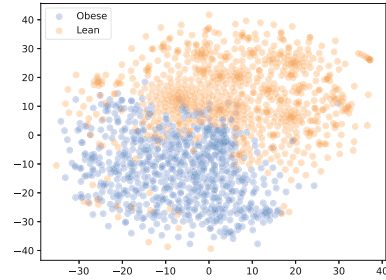
Methods	Subject-level scores	Epoch-level scores
1-D CNN	0.951 \pm 0.104	0.937 \pm 0.130
SVM	0.917 \pm 0.170	0.920 \pm 0.139
KNN	0.903 \pm 0.135	0.909 \pm 0.143
RF	0.936 \pm 0.152	0.937 \pm 0.163
ADB	0.883 \pm 0.299	0.898 \pm 0.236

Table 2. Test scores (mean accuracy \pm standard deviation) of benchmark models and the proposed model.

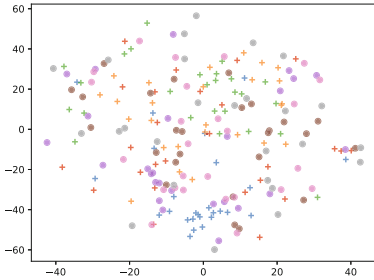
Methods	Subject-level scores	Epoch-level scores
EEGNet	0.610 \pm 0.162	0.578 \pm 0.137
Shallow ConvNet	0.670 \pm 0.132	0.631 \pm 0.114
Deep ConvNet	0.670 \pm 0.151	0.582 \pm 0.133
A-VGGRI	0.742 \pm 0.138	0.798 \pm 0.113
MuLHiTA	0.798 \pm 0.169	0.814 \pm 0.127
AgeNet	0.817 \pm 0.166	0.587 \pm 0.060
ParkinsonNet	0.583 \pm 0.118	0.583 \pm 0.071
CognitionNet	0.567 \pm 0.110	0.589 \pm 0.057
Proposed Model	0.951 \pm 0.110	0.937 \pm 0.134



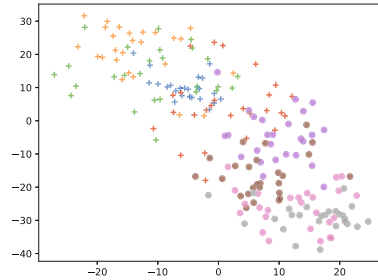
(a) Visualization of features learned by EEGNet.



(b) Visualization of features learned by VAE.



(c) Subject-level view of features learned by EEGNet.



(d) Subject-level view of features learned by VAE.

Fig. 3. 2-D visualizations of learned feature representations using t-SNE. Top row: (a) Features learned by EEGNet at its spatial convolution layer and (b) features learned by VAE. Bottom row: (c) Features from 8 randomly selected subjects (4 obese marked with ‘+’ and 4 lean marked with ‘o’) learned by EEGNet, and (d) Features learned by VAE. Each subject’s data points are coloured consistently. The VAE feature set shows distinct subject-based clusters.

3.2 Evaluation of Feature Representations

To illustrate the learned feature representations, we applied t-distributed stochastic neighbour embedding (t-SNE) [30], a non-linear dimensionality reduction technique, to project the feature representations learned by EEGNet at the spatial convolution layer and our proposed VAE into 2-D Euclidean spaces. We chose t-SNE for its ability to effectively preserve the local structure of high-dimensional data by minimizing differences between the joint distributions of high-dimensional and low-dimensional data [31].

We visualized the feature representations in two scenarios: across all subjects and on a subset of 8 randomly selected subjects. Figure 3a shows the feature representation learned by EEGNet, and Fig. 3b shows the feature representation learned by the proposed VAE across all subjects. Similarly, Fig. 3c shows the

features learned by EEGNet, and Fig. 3d shows the features learned by the proposed VAE on the subset of 8 randomly selected subjects. These visualizations indicate that features learned by the proposed VAE exhibit a distinct distribution between the obese and lean groups, demonstrating effective separation of underlying data patterns.

One potential explanation for the between-group discrimination contained in the feature representations learned by the proposed VAE, which contributes to the significantly enhanced performance of our model, lies in the nature of resting-state EEG data. Resting-state EEG data inherently contains differences between subjects. These differences can be regarded as irrelevant noise in the context of the classification task [32]. Therefore, a model capable of learning high-level features that incorporate subject-specific distinctions allows it to filter out the irrelevant variations or noise caused by these differences between subjects, thereby enhancing its ability to perform accurate classifications based on relevant patterns in the EEG data. This relationship is visually demonstrated in Fig. 3c and Fig. 3d, which present the learned feature representations organized by subject. Moreover, we infer that supervised feature learning adjusts the learned features based on the gradient outcomes from the classification process. In contrast, as an unsupervised model, VAE primarily focuses on reconstructing the input data, which reduces the risk of overfitting during training. Consequently, VAE is more likely to learn diverse and adaptable feature representations.

Additionally, we applied the proposed DI measure to the learned feature representations of EEGNet and the proposed VAE, resulting in $DI = 0.247$ for EEGNet features and $DI = 0.220$ for VAE features. Given that effective features typically contribute significantly to classifier performance, we compared the first quartile of these feature sets-specifically, the top 25% with the lowest DI_i values—as depicted in Fig. 4. The DI values associated with the VAE features exhibit a more distinct separation compared to those of EEGNet, indicating the good discriminative quality of the VAE features.

3.3 Analysis of Channel Importance for Obesity

We computed the average output values from the spatial convolution layer of the VAE encoder to assess the importance of individual EEG channels in relation to obesity. Figure 5a visually represents these values (a larger value is represented with a higher colour intensity). Higher average output values for a channel in the lean group indicate stronger responses to EEG signals typical of lean individuals, highlighting pronounced EEG patterns relevant to lean physiology. Conversely, higher values in the obese group denote stronger responses to EEG signals characteristic of obese individuals, emphasizing distinct EEG patterns associated with obesity. The contrasting colour patterns in Fig. 5a (e.g., dark red versus dark blue) indicate opposite contributions to the VAE’s feature representation.

To provide a more intuitive demonstration of channel importance, we computed the absolute difference between the importance of each channel for lean

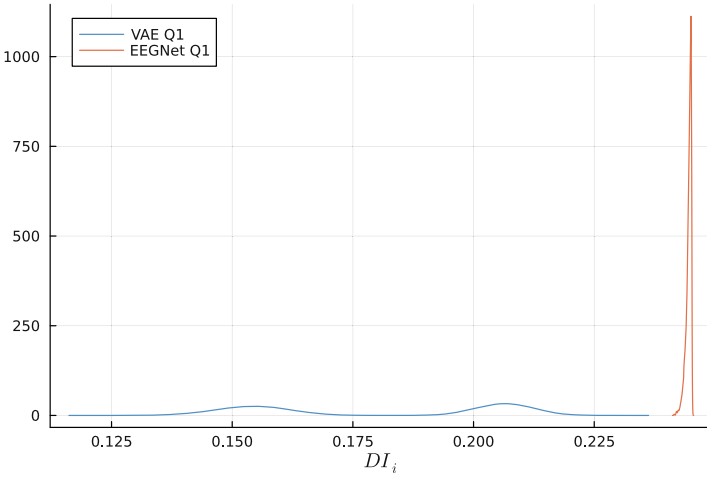
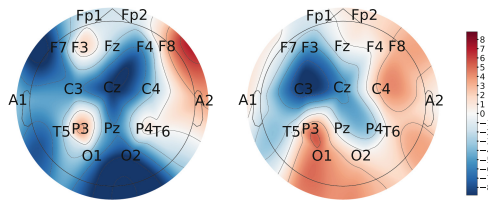
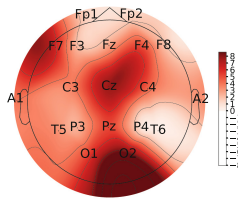


Fig. 4. Comparison of the first quantile of the DI scores from feature representations learned by EEGNet and VAE.



(a) Channel importance for lean group (left) and obese group (right). Higher colour intensity indicates higher channel importance.



(b) Absolute (i.e. ‘net’) channel importance. Higher colour intensity indicates higher channel importance.

Fig. 5. Visualization of Obesity-related Channel Importance.

and obese groups, referred to as the ‘net’ channel importance. This is demonstrated in Fig. 5b, where darker colours indicate higher ‘net’ channel importance. The absolute value of this difference, or the ‘net’ channel importance, reflects the overall contribution of each EEG channel in distinguishing between lean and obese individuals. Higher values signify channels that play a more significant role in capturing EEG patterns specific to either lean or obese individuals. This analysis helps in identifying which EEG channels are most crucial for differentiating between these two groups based on their neural activity patterns.

4 Ablation Study

We conducted an ablation study to assess the contribution of the VAE feature learning process. Using the best-performing 1-D CNN classification algorithm, we compared the classification performance on preprocessed EEG signals alone and on the feature representations learned via VAE. The results demonstrated that using only preprocessed EEG signals achieved accuracies of 0.54 for sample-level and 0.55 for subject-level classification. In contrast, incorporating VAE-learned features significantly improved accuracies to 0.94 and 0.95, respectively, highlighting the substantial impact of the VAE feature learning process on enhancing classification performance.

5 Conclusion

This study investigated obesity-related brain activities using resting-state EEG data through a DL approach. We employed a VAE to learn latent feature representations from EEG signals, followed by classification using a 1-D CNN. Our analysis highlights distinct spatial patterns that differentiate obese from lean brains, providing insights into neural activity associated with obesity.

In future research, we aim to extend our investigation to explore obesity patterns of male individuals. This will involve expanding our dataset to encompass male subjects and adjusting our model accordingly to provide a comprehensive analysis of obesity across both sexes. Additionally, we will focus on enhancing model interpretability by incorporating discussions on temporal and spatial information, thereby advancing our understanding of obesity-related brain dynamics.

References

1. Buechler, C., Wanninger, J., Neumeier, M.: Adiponectin, a key adipokine in obesity related liver diseases. *World J. Gastroenterol. WJG* **17**(23), 2801–2811 (2011). <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3120939/>
2. Silva, G.B.D., Bentes, A.C.S.N., Daher, E.D.F., Matos, S.M.A.D.: Obesity and kidney disease. *Braz. J. Nephrol.* **39**, 65–69 (2017)

3. Verma, S., Hussain, M.E.: Obesity and diabetes: an update. *Diab. Metabolic Syndr. Clin. Res. Rev.* **11**(1), 73–79 (2017). <https://www.sciencedirect.com/science/article/pii/S1871402116300662>
4. Wolin, K.Y., Carson, K., Colditz, G.A.: Obesity and cancer. *Oncologist* **15**(6), 556–565 (2010). <https://doi.org/10.1634/theoncologist.2009-0285>
5. Włodarczyk, M., Nowicka, G.: Obesity, DNA damage, and development of obesity-related diseases. *Int. J. Molec. Sci.* **20**(5), 1146 (2019). <https://www.mdpi.com/1422-0067/20/5/1146>
6. Sui, S.X., Pasco, J.A.: Obesity and brain function: the brain-body crosstalk. *Medicina* **56**(10), 499 (2020). <https://www.mdpi.com/1648-9144/56/10/499>
7. Lowe, C.J., Reichelt, A.C., Hall, P.A.: The prefrontal cortex and obesity: a health neuroscience perspective. *Trends Cogn. Sci.* **23**(4), 349–361 (2019). <https://www.sciencedirect.com/science/article/pii/S1364661319300221>
8. Bethge, D., et al.: EEG2Vec: learning affective EEG representations via variational autoencoders. In: 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pp. 3150–3157 (2022). iSSN: 2577-1655
9. Babiloni, C., et al.: Classification of single normal and alzheimer’s disease individuals from cortical sources of resting state EEG rhythms. *Front. Neurosci.* **10** (2016). <https://www.frontiersin.org/article/10.3389/fnins.2016.00047>
10. Allison, B.Z., Wolpaw, E.W., Wolpaw, J.R.: Brain–computer interface systems: progress and prospects. *Expert Rev. Med. Dev.* **4**(4), 463–474 (2007). <https://doi.org/10.1586/17434440.4.4.463>
11. DiFeliceantonio, A.G., Small, D.M.: Dopamine and diet-induced obesity. *Nat. Neurosci.* **22**(1), 1–2 (2019). <https://www.nature.com/articles/s41593-018-0304-0>
12. Yue, Y., De Ridder, D., Manning, P., Ross, S., Deng, J.D.: Finding neural signatures for obesity through feature selection on source-localized EEG (2022). [arXiv:2208.14007](https://arxiv.org/abs/2208.14007)
13. O’Brien, P.D., Hinder, L.M., Callaghan, B.C., Feldman, E.L.: Neurological consequences of obesity. *Lancet Neurol.* **16**(6), 465–477 (2017). <https://www.sciencedirect.com/science/article/pii/S1474442217300844>
14. Blume, M., Schmidt, R., Hilbert, A.: Abnormalities in the eeg power spectrum in bulimia nervosa, binge-eating disorder, and obesity: a systematic review. *Eur. Eat. Disord. Rev.* **27**(2), 124–136 (2019)
15. Imperatori, C., et al.: Modification of EEG functional connectivity and EEG power spectra in overweight and obese patients with food addiction: an eLORETA study. *Brain Imaging Behav.* **9**(4), 703–716 (2015). <https://doi.org/10.1007/s11682-014-9324-x>
16. De Ridder, D., et al.: The brain, obesity and addiction: an eeg neuroimaging study. *Sci. Rep.* **6**(1), 34122 (2016)
17. Wang, Z., Wang, Y.: Extracting a biologically latent space of lung cancer epigenetics with variational autoencoders. *BMC Bioinf.* **20**, 1–7 (2019)
18. Ahmed, T., Longo, L.: Examining the size of the latent space of convolutional variational autoencoders trained with spectral topographic maps of EEG frequency bands. *IEEE Access* **10**, 107575–107586 (2022)
19. Dong, C., Xue, T., Wang, C.: The feature representation ability of variational autoencoder. In: 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC), pp. 680–684. IEEE (2018)
20. Horstmann, A., et al.: Obesity-related differences between women and men in brain structure and goal-directed behavior. *Front. Hum. Neurosci.* **5**, 58 (2011)

21. Lovejoy, J.C., Sainsbury, A., Stock Conference 2008 Working Group.: Sex differences in obesity and the regulation of energy homeostasis. *Obesity Rev.* **10**(2), 154–167 (2009)
22. Coveleskie, K., et al.: Altered functional connectivity within the central reward network in overweight and obese women. *Nutr. Diab.* **5**(1), e148–e148 (2015)
23. Lawhern, V.J., Solon, A.J., Waytowich, N.R., Gordon, S.M., Hung, C.P., Lance, B.J. : EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces. *J. Neural Eng.* 15(5), 056013(2018). <https://iopscience.iop.org/article/10.1088/1741-2552/aace8c>
24. Biró, T.S., Néda, Z.: Gintropy: gini index based generalization of entropy. *Entropy* **22**(8), 879 (2020)
25. Schirrneister, R.T., et al.: Deep learning with convolutional neural networks for eeg decoding and visualization. *Hum. Brain Mapp.* **38**(11), 5391–5420 (2017)
26. Xia, L., et al.: MuLHiTA: a novel multiclass classification framework with multi-branch LSTM and hierarchical temporal attention for early detection of mental stress. *IEEE Trans. Neural Netw. Learn. Syst.* **34**(12), 9657–9670 (2022)
27. Khayretdinova, M., Shovkun, A., Degtyarev, V., Kiryasov, A., Pshonkovskaya, P., Zakharov, I.: Predicting age from resting-state scalp eeg signals with deep convolutional neural networks on td-brain dataset. *Front. Aging Neurosci.* **14**, 1019869 (2022)
28. Li, K., Ao, B., Wu, X., Wen, Q., Ul Haq, E., Yin, J.: Parkinson’s disease detection and classification using EEG based on deep CNN-LSTM model. *Biotechnol. Genet. Eng. Rev.*, 1–20 (2023)
29. Sibilano, E., et al.: An attention-based deep learning approach for the classification of subjective cognitive decline and mild cognitive impairment using resting-state eeg. *J. Neural Eng.* **20**(1), 016048 (2023)
30. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. *J. Mach. Learn. Res.* **9**(11) (2008)
31. Birjandtalab, J., Pouyan, M.B., Nourani, M.: Nonlinear dimension reduction for EEG-based epileptic seizure detection. In: 2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI), pp. 595–598 (2016). ISSN: 2168-2208
32. Bijsterbosch, J., Harrison, S., Duff, E., Alfaro-Almagro, F., Woolrich, M., Smith, S.: Investigations into within-and between-subject resting-state amplitude variations. *Neuroimage* **159**, 57–69 (2017)



A Deep Learning System for Water Pollutant Detection Based on the SENSIPLUS Microsensor

Hamza Mustafa^{1(✉)}, Mario Molinara¹, Luigi Ferrigno¹, and Michele Vitelli²

¹ Department of Electrical and Information Engineering, University of Cassino and Southern Lazio, 03043 Cassino, Italy

{hamza.mustafa,m.molinara,ferrigno}@unicas.it

² Sensichips s.r.l., 04011 Aprilia, Italy

michele.vitelli@sensichips.com

Abstract. Accurate classification of water pollutants is paramount for safeguarding the environment. This study presents an innovative approach to classifying water pollutants by integrating deep learning algorithms with effective preprocessing techniques. The Sensichips Smart Water Cable Sensor (SCW) facilitates real-time data acquisition for various water pollutants, establishing a robust foundation for comprehensive analysis. SCW utilizes the SENSIPLUS chip, which employs impedance spectroscopy to detect a variety of water-soluble pollutants via an array of sensors. Our research adopts an end-to-end sensor-to-classification framework, leveraging deep neural networks to capture temporal data dynamics. Employing a CNN model with a sliding window approach, our method demonstrates promising results, achieving an average accuracy of 95.78% across ten folds for classifying eight distinct water pollutants. The low-cost IoT-based infrastructure makes this approach scalable and accessible for deployment in water monitoring systems.

Keywords: Deep Learning · Smart Cable Water · Smart Sensors · Water Pollutant Detection

1 Introduction

Water pollution is a pressing issue that can have devastating effects on both the environment and human health. The contamination of water bodies by various pollutants, such as industrial chemicals and waste, agricultural runoff, and household waste, threatens aquatic life, plants, and humans. Additionally, it contributes to the high prevalence of water-borne illnesses, among the top 10 global causes of mortality [22]. Identifying pollutants is essential to monitoring water quality, which is important for environmental management and protecting water resources. Despite the availability of modern, advanced technology, most drinking water production plants in developing countries continue to rely on conventional techniques for identifying pollutants in water. The presence of

pollutants is detected by conducting a chemical analysis in the laboratory. One drawback of these technologies is their reliance on human expertise and considerable time investment. Additionally, these systems suffer from extended delay times, hindering close real-time water quality monitoring. Monitoring stations increasingly utilize advanced technologies like remote sensing (RS) and the Internet of Things (IoT) to address these challenges, rapidly generating vast amounts of data.

Conventional machine learning techniques cannot perform well on large data. In recent years, Deep Learning (DL) has shown remarkable success in various domains, including image recognition, natural language processing, and speech recognition [17]. Its ability to automatically learn intricate patterns and representations from data makes DL well-suited for complex and high-dimensional datasets. Long Short-Term Memory Recurrent Neural Networks (LSTM RNNs) have been widely used in time series classification due to their effectiveness in handling temporal dependencies. This study presents a new methodology for classifying water pollutants by integrating cutting-edge technology and data processing techniques optimized for deep learning models. We use the Sensichips Smart Cable Water(SCW) based on SENSIPLUS microsensors, a low-power, low-cost, and real-time solution to gather pollutant data in water. SENSIPLUS is a proprietary technology of Sensichips s.r.l. developed with the University of Pisa [20]. This facilitates the development of a more refined understanding of water quality dynamics.

This paper is structured as follows: Sect. 2 discusses the related work, providing a review of previous studies. Section 3 presents the proposed approach, detailing the methodology and data preprocessing techniques used. Results are presented in Sect. 4. Finally, Sect. 5 concludes the paper by summarizing the main findings and suggesting avenues for future research.

2 Related Work

Machine learning (ML) approaches have recently gained popularity for evaluating and classifying water quality. [19] employed various machine learning models, including SVM, Random Forest, and others, to classify water quality based on the water quality index (WQI). The study conducted by [11] utilized the Random Forest classifier to categorize and predict water quality. [8] employed LSTM RNNs and SVM to classify water quality into three categories using physicochemical data. In their work, [14] introduced a decision tree model aimed at predicting six different quality indicators: pH, temperature, chemical oxygen demand (COD), ammonia-nitrogen (NH₃-N), nitrate-nitrogen (NO₃-N), and pH. SVM, artificial neural networks (ANN), and group method of data handling (GMDH) are the three ML algorithms that were compared by [13] in order to estimate the water quality of the Tireh River in southwest Iran.

However, all the above techniques discussed are classifying water quality. Based on our present knowledge, only a limited number of methods use sensor technology and deep learning to classify pollutants. [1] employs an IoT-based

system integrated with KNN and SVM algorithms to monitor water pollutants, including pH, temperature, conductivity, and turbidity. The study is limited to general water pollution detection, such as turbidity, and not specific pollutants in water. [16] developed a Water Quality Monitoring (WQM) system using a single-chip solution that incorporates FPGA technology and wireless connectivity via an XBee module. The system tracks and measures pH, temperature, humidity, turbidity, carbon dioxide, and water levels. However, this approach is limited to the above-mentioned quality parameters, not the pollutants specifically. [23] use a Conv.LSTM network to classify water pollution by monitoring nitrate, dissolved oxygen, conductivity, biological oxygen demand, total coliform, and fecal coliform. This study is limited to the mentioned parameters no other pollutants were classified. [15] develops a water pollution monitoring system using the behavior of *Caenorhabditis elegans*, LSTM models to analyze time-series data from nematode movements. The system as detailed primarily focuses on detecting formaldehyde and benzene, with less emphasis on a broad spectrum of pollutants. [21] introduce a low-cost sensing platform using natural language generation to classify wastewater pollutants, significantly improving detection accuracy over traditional methods. The nitrate content in groundwater is estimated by the authors in [5] using prial component regression and artificial neural networks. This study is limited to only nitrate content estimation no other pollutants were estimated. The authors use partial least square discriminant analysis in [7] to find explosive components in sewage water. To identify and classify the chemicals found in seawater, CNN and LSTM were used by [6]. [2] employs machine learning techniques to leverage non-changeable factors such as latitude, longitude, and elevation, predicting pH, temperature, turbidity, dissolved oxygen hardness, chlorides, alkalinity, and chemical oxygen demand in water bodies, pioneering a novel approach to predicting water contamination. Their approach is only limited to the prediction of the aforementioned factors. Some of the authors developed systems able to monitor both water and air thanks to the SENSIPLUS platform [3, 4, 9, 18].

In our previous work [10], an artificial neural network with one hidden layer was used to classify 5 pollutants in the water. The SENSIPLUS chip was used for the data collection. KNN and SWM along with the anomaly detection algorithms were used to detect pollutants in the water in another of our previous works [12]. Table 1 compares the different approaches for water pollutants and quality classification.

3 Methodology

Data collection, preprocessing, and implementing deep learning algorithms comprise the three main phases of our proposed method. Classifying the pollutants found in water is our primary goal. Sliding window methodology is employed to prepare the input data for classification. Presently, eight distinct chemicals that are often found in water samples are being classified using our proposed method. Sensichips s.r.l.'s Smart Cable Water (SCW) Sensor collected pollution

Table 1. Comparison of studies on water pollutants detection

Ref	Methodology	Data Type	Models/Technologies	Key Findings/Limitations
[1]	IoT-based water pollutant monitoring	pH, temperature, etc.	KNN, SVM	Focuses on general pollution detection, not specific pollutants
[2]	ML for predicting water contamination factors	Various water parameters	ML techniques	Limited to non-changeable geographical factors
[5]	Estimation of nitrate in groundwater	Nitrate content	ANN, PCR	Limited to nitrate content estimation
[6]	Classification of seawater chemicals	Seawater chemicals	CNN, LSTM	Focus on specific seawater chemicals
[7]	Detection of explosives in sewage	Explosive components	PLS-DA	Specific to explosive components in sewage
[8]	Classification of water quality into categories	Physicochemical data	LSTM, SVM	Categorizes water quality into three types
[10]	Classification of pollutants in water	Various pollutants	ANN	Focus on five specific pollutants
[11]	Prediction and categorization of water quality	Water quality data	Random Forest	Employed for categorization and prediction
[12]	Detection of water pollutants	Various pollutants	KNN, SVM, Anomaly Detection	Detection focused on specific pollutants
[13]	Estimation of water quality, Tیره River	Water quality data	SVM, ANN, GMDH	Comparison of three ML models
[14]	Prediction of water quality indicators	pH, COD, etc.	Decision Trees	Predicts six water quality indicators
[15]	Monitoring system for specific pollutants	Nematode behavior data	LSTM	Targets formaldehyde and benzene detection
[16]	Water Quality Monitoring (WQM) system	Various water parameters	FPGA, XBee	Limited to basic quality parameters, not pollutants
[19]	Classification based on WQI	Water quality index data	SVM, Random Forest	Focus on WQI-based classification
[21]	Detection of water pollutants	Various pollutants	NLP	Focus on specific water pollutants
[23]	Classification of specific water pollutants	Various pollutants	Conv.LSTM	Focus on specific water pollutants

data. The SCW is built on the SENSIPLUS framework and consists of InterDigitated Electrodes (IDEs). With its versatile and accurate Electrical Impedance Spectrometer (EIS), the SENSIPLUS micro-chip can assess both on-chip and off-chip sensors throughout a frequency range of 3.1 MHz to 1.2 MHz. It is possible to do measurements with the SENSIPLUS by using a variety of sensors. In particular, the six IDEs that make up the SCW system are metalized with silver, copper, platinum, palladium, nickel, silver, and gold.

The following eight pollutants data were collected:

- Acetic Acid
- Ammonia
- Hydrogen Peroxide
- Hydrochloric Acid
- Phosphoric Acid
- Sodium Chloride
- Sodium Hydroxide
- Sodium Hypochlorite

These pollutants originate from various industrial, agricultural, and residential activities, making their presence ubiquitous and monitoring crucial for environmental and human health management. Each pollutant poses distinct health risks upon exposure, ranging from respiratory irritation to severe burns and long-term health effects. The method used to measure each pollutant consists of two phases. Firstly, 600 potable water samples are collected at a rate of 0.5 Hz to stabilize all the sensors. Measurement of the pollutants is the next step. After 600 samples, the pollutant is injected slowly into the water, and a thousand samples are taken to record the growth of the whole sensor. One of the main challenges in machine learning is identifying and choosing key sensor features to improve classification accuracy. To acquire a 10-size feature vector, the appropriate features are gathered:

- At a frequency of 78 kHz, the impedance of IDEs made of gold and platinum was measured.
- At a frequency of 200 Hz, the impedance of IDEs made of gold, platinum, silver, and nickel was measured.

By obtaining 10 sets of 1600 samples for each contaminant (which included potable water) and using the previously indicated measurement technique, a total of 144000 samples were collected.

3.1 Data Preprocessing

The raw data is initially analyzed using the EMA technique. The data's EMA was estimated using the Eq. 1.

$$E_{EMA_i} = \left(\frac{a-1}{a}\right) E_{EMA_{i-1}} + \left(\frac{1}{a}\right) x_{EMA_i} \quad (1)$$

Then, the differences between successive EMA values are computed from the original data (OD) for each data point. These differences, called EMA Differences (EMA-D), highlight deviations or changes from the underlying patterns in the data. By emphasizing these variations, the classification method can better capture and understand the data dynamics, which is crucial for accurately distinguishing between different classes or categories. Equation 2 is used to compute the differences.

$$EMA - D = OD - E_{EMA_i} \tag{2}$$

Then, we filter out the initial transient data and retain only the stable state data points. By filtering out initial transient data, the model focuses on learning from stable state data points, thus reducing the influence of temporary fluctuations and noise. We discard the initial 400 samples from each measure and then implement a sliding window technique on the EMA Differences (EMA-D) data. Using the sliding window (SW) approach, data was divided into overlapping windows, as shown in Fig. 1. The SW of size 32 sequentially traverses the data, shifting the sliding window one step at every iteration. The model uses a collection of input sequences generated via this methodology. Through the implementation of this approach, it is assumed that the model will possess the ability to learn trends and data correlations. Figure 2 presents a graphical representation of one thousand data samples acquired during acetic acid measurements by the SCW utilizing one of its IDEs (OFFCHIP NICKEL 200 Hz IN-PHASE).

3.2 Pollutants Classification

We used CNNs, LSTM networks, and Simple RNNs to classify pollutants. The CNN architecture has been tailored to classify pollutants into one of the eight distinct classes. CNNs are often used for image-related tasks, such as processing images using RGB channels, which consist of three channels in the input tensor. However, the water pollution data for our problem are provided in a time series

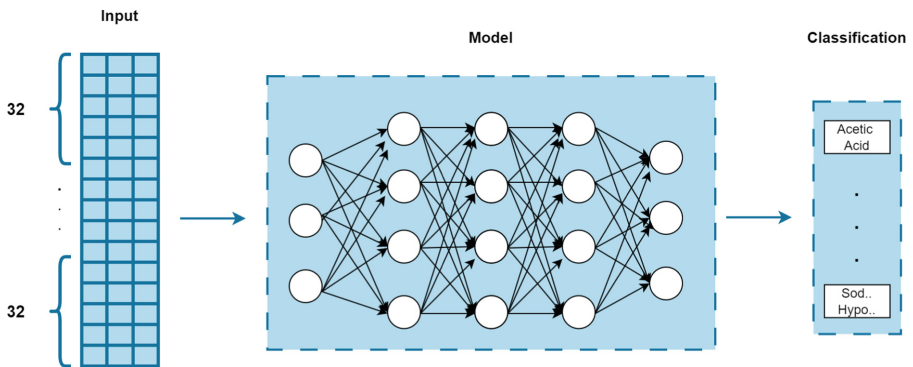


Fig. 1. Proposed Approach

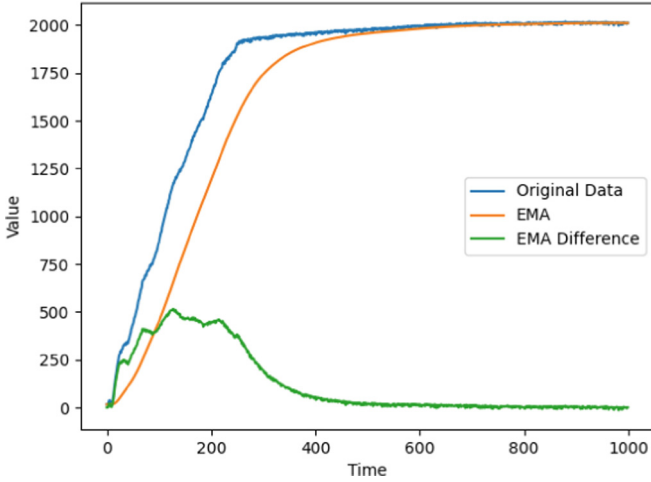


Fig. 2. A graphic illustration of the original data, EMA, and difference values related to Acetic Acid

manner. The proposed method divides the time series data into windows using the sliding windows approach. This window serves as the input tensor for the model, allowing it to capture dependencies and extract features.

The total number of windows was calculated as follows: $N - W + 1$ windows were obtained by subtracting the window size (W) from the total number of data points (N) and adding one. A subset of the EMA-D sequence consisting of W consecutive data points, each having F features (sensor measure), is represented by each window. X_i was the identifier for these windows, and i refers to any integer between 1 and the total number of windows. As an example, X_i was represented as a matrix in which a feature was represented by each column and a data point inside the window by each row.

$$Y_i = \begin{bmatrix} y_{i,1} & y_{i,2} & \dots & y_{i,F} \\ y_{i+1,1} & y_{i+1,2} & \dots & y_{i+1,F} \\ \vdots & \vdots & \ddots & \vdots \\ y_{i+W-1,1} & y_{i+W-1,2} & \dots & y_{i+W-1,F} \end{bmatrix}$$

Let $y_{i,j}$ be the j^{th} feature of the i^{th} data point in the window. Temporal dependencies were captured via overlapping windows, where the amount of overlap was specified by the stride (S_t). Using the sliding window approach, we successfully divided the EMA-D sequences into segments, which aided in capturing crucial temporal trends and dependencies required for training the CNN model. The CNN model architecture is specifically designed to handle a feature set consisting of 10 distinct features. These features are initially organized into a tensor size 32×10 by applying a sliding window size of 32. This tensor serves as

the input data for the subsequent model layers. The architecture and the layers of the CNN model used are shown in Table 2.

Table 2. CNN Architecture

Lyer Type	Configuration
Input	Input shape: (height, width, channels)
Convolutional	Filters: 32, Kernel size: (3, 3)
Activation	ReLU
Batch Normalization	
Convolutional	Filters: 64, Kernel size: (3, 3)
Activation	ReLU
Batch Normalization	
Dropout	Dropout rate: 0.5
Max Pooling	Pool size: (1, 1)
Convolutional	Filters: 128, Kernel size: (3, 3)
Activation	ReLU
Batch Normalization	
Dropout	Dropout rate: 0.5
Convolutional	Filters: 256, Kernel size: (3, 3)
Activation	ReLU
Batch Normalization	
Dropout	Dropout rate: 0.5
Flatten	
Dense	Units: 128, Activation: ReLU
Dense	Units: 64, Activation: ReLU
Output (Dense)	Units: Number of classes, Activation: Softmax

This comprehensive architecture, optimized using stochastic gradient descent, enables the accurate classification of pollutants based on the feature-rich EMA-D sequences derived from the initial 32×10 tensor. On the other hand, LSTM networks and Simple RNN are employed due to their ability to identify temporal correlations in sequential data. Transformers models were also used to classify the pollutants, but they didn't perform well. The input of these models is the same as the CNN. To comprehensively evaluate the performance of all the models, a 10-fold cross-validation technique is utilized. Each model undergoes ten repetitions of training and validation, following which the dataset is divided into ten unique subsets. Nine subsets are used for training in each cycle, with one subset designated as the validation set. The process is carried out on each of the 10 folds to ensure that each data point is included in the set of validation data at least once.

4 Results

The classification performance of the Transformers, RNN, LSTM, and CNN networks across ten-fold cross-validation is shown in Table 3. For every fold, the classification accuracy and the mean accuracy are presented.

Table 3. Classification Performance

Folds	Transformers	RNN	LSTM	CNN
Fold-1	47.33%	75.43%	83.88 %	85.58%
Fold-2	40.40%	79.53%	91.15 %	94.21%
Fold-3	50.18%	92.42%	97.28%	97.48%
Fold-4	47.31%	88.04%	97.18 %	99.60%
Fold-5	57.54%	89.47%	97.69 %	98.26%
Fold-6	61.51%	83.90%	97.94 %	97.94%
Fold-7	74.86%	94.33%	99.39 %	99.23%
Fold-8	79.63%	85.21%	88.16 %	95.37%
Fold-9	71.74%	82.91%	88.43 %	92.89%
Fold-10	56.63%	81.64%	95.50 %	97.26%
Mean Accuracy	58.68%	85.28%	93.66%	95.78%

The mean classification accuracies for Transformers RNN, LSTM, and CNN are 58.68%, 85.28%, 93.66%, and 95.78%, respectively. These results indicate that CNN is the best-performing model, followed by LSTM, RNN, and transformers being the least accurate. The CNN model attained a mean classification accuracy of 95.56%. The model maintained excellent accuracy throughout numerous folds, with a peak of 99.60% in Fold-4. As depicted in the confusion matrix for Fold-4 (Fig. 3) of CNN model, only Acetic Acid exhibits a slight confusion with Phosphoric Acid, while all other pollutants are accurately classified. Additionally, LSTM outperformed RNN, achieving a mean accuracy of 93.66%. The performance of RNN, compared to the other two networks, is not good since RNN networks are affected by the issue of vanishing gradients during training. Transformers, with a mean accuracy of 58.68%, demonstrate the lowest performance among the models. With a marginal difference of roughly 2% the performance of LSTM and CNN is almost the same.

Table 4. Mean and Standard Deviation(STD) of Classification Accuracies

Model	Mean	STD
Transformers	58.68%	14.38%
RNN	85.28%	6.01%
LSTM	93.66%	4.73%
CNN	95.78%	2.27%

One critical aspect to consider is the stability of the models across different folds. The standard deviation of the accuracies, as shown in Table 4, indicates this stability. CNNs exhibit the lowest standard deviation (2.27%), underscoring their robust performance and consistent accuracy across all folds. This consistency can be attributed to the CNN’s ability to capture local spatial features effectively within the sliding windows, making it highly suitable for this classification task. LSTMs follow with a standard deviation of 4.73%, indicating reliable performance but slightly less stability than CNNs. RNNs show moderate variability with a standard deviation of 6.01%, reflecting their susceptibility to the vanishing gradient problem. Transformers, with the highest standard deviation of 14.38%, demonstrate significant fluctuations, suggesting that their performance is less stable and consistent in this context. However, the limited window size may hinder the full potential of Transformers, which typically excels in larger contexts. Overall, CNNs emerge as the most suitable model for classifying signals using a sliding window approach, followed closely by LSTMs.

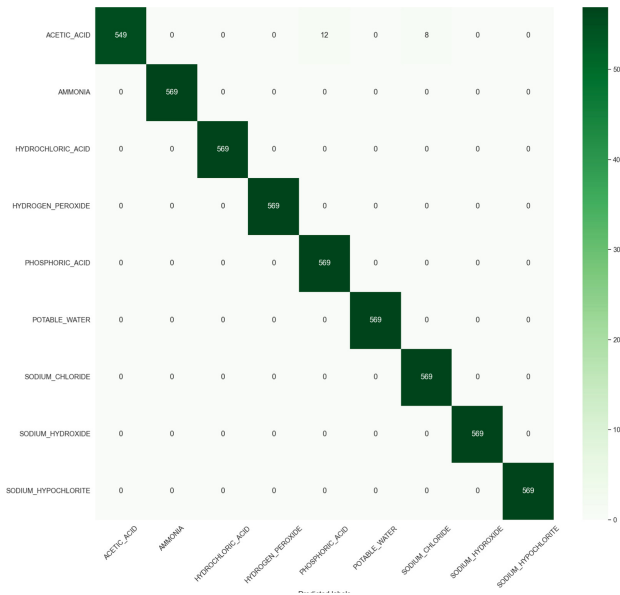


Fig. 3. CNN Fold-4 Confusion Matrix

5 Conclusion

In conclusion, in this study by leveraging the capabilities of the Sensichips Smart Water Cable Sensor and implementing deep learning models, we have developed an efficient system for classifying the eight most commonly found pollutants in water. By, employing Convolutional Neural Networks (CNN), our study

attained an impressive classification accuracy rate of 95.56%. The accuracy of our technique showcases its ability to significantly improve the precision of classifying pollutants in water, hence potentially having a significant influence on water quality monitoring. Incorporating the sliding window technique enhances our method's ability to efficiently capture temporal relationships, constituting one of its key advantages. Accurately classifying pollutants in water is a continuous task. Consistent improvement and advancement are essential to effectively address the growing requirements of water pollutant evaluation. However, our study is limited to eight pollutants, and the model does not perform well with the transient data included. Future research directions involve incorporating additional pollutant measurement data and enhancing the model's ability to handle non-steady state data by exploring various deep learning models, such as Convolutional Long Short-Term Memory (ConvLSTM). Additionally, ensemble approaches can be utilized to integrate classification capabilities from multiple models.

References

1. AlZubi, A.A.: Iot-based automated water pollution treatment using machine learning classifiers. *Environ. Technol.* **45**(12), 2299–2307 (2024)
2. Banerjee, K., Bali, V., Nawaz, N., Bali, S., Mathur, S., Mishra, R.K., Rani, S.: A machine-learning approach for prediction of water contamination using latitude, longitude, and elevation. *Water* **14**(5), 728 (2022)
3. Bourelly, C., et al.: A preliminary solution for anomaly detection in water quality monitoring. In: 2020 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 410–415 (2020)
4. Bria, A., Cerro, G., Ferdinandi, M., Marrocco, C., Molinara, M.: An iot-ready solution for automated recognition of water contaminants. *Pattern Recogn. Lett.* **135**, 188–195 (2020)
5. Charulatha, G., Srinivasalu, S., Uma Maheswari, O., Venugopal, T., Giridharan, L.: Evaluation of ground water quality contaminants using linear regression and artificial neural network models. *Arab. J. Geosci.* **10**, 1–9 (2017)
6. Dean, S.N., Shriver-Lake, L.C., Stenger, D.A., Erickson, J.S., Golden, J.P., Trammell, S.A.: Machine learning techniques for chemical identification using cyclic square wave voltammetry. *Sensors* **19**(10), 2392 (2019)
7. Desmet, C., Degiuli, A., Ferrari, C., Romolo, F.S., Blum, L., Marquette, C.: Electrochemical sensor for explosives precursors' detection in water. *Challenges* **8**(1), 10 (2017)
8. Dilmi, S., Ladjal, M.: A novel approach for water quality classification based on the integration of deep learning and feature extraction techniques. *Chemom. Intell. Lab. Syst.* **214**, 104329 (2021)
9. Ferdinandi, M., et al.: A novel smart system for contaminants detection and recognition in water. In: 2019 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 186–191 (2019)
10. Ferdinandi, M., et al.: A novel smart system for contaminants detection and recognition in water. In: 2019 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 186–191. IEEE (2019)

11. Flores, V., Bravo, I., Saavedra, M.: Water Quality classification and machine learning model for predicting water quality status—a study on loa river located in an extremely arid environment: atacama desert. *Water* **15**(16), 2868 (2023)
12. Gerevini, L., et al.: An end-to-end real-time pollutants spilling recognition in wastewater based on the iot-ready sensiplus platform. *J. King Saud Univ.-Comput. Inf. Sci.* **35**(1), 499–513 (2023)
13. Haghbi, A.H., Nasrolahi, A.H., Parsaie, A.: Water quality prediction using machine learning methods. *Water Qual. Res. J.* **53**(1), 3–13 (2018)
14. Jaloree, S., Rajput, A., Gour, S.: Decision tree approach to build a model for water quality. *Binary J. Data Mining Network.* **4**(1), 25–28 (2014)
15. Kang, S.-H., Jeong, I.-S., Lim, H.-S.: A deep learning-based biomonitoring system for detecting water pollution using *caenorhabditis elegans* swimming behaviors. *Eco. Inf.* **80**, 102482 (2024)
16. Konde, S., Deosarkar, S.: Iot based water quality monitoring system. In: 2nd International Conference on Communication & Information Processing (ICCIP) (2020)
17. Mohammadi, M., Al-Fuqaha, A., Sorour, S., Guizani, M.: Deep learning for iot big data and streaming analytics: a survey. *IEEE Commun. Surv. Tutor.* **20**(4), 2923–2960 (2018)
18. Molinara, M., Ferdinandi, M., Cerro, G., Ferrigno, L., Massera, E.: An end to end indoor air monitoring system based on machine learning and sensiplus platform. *IEEE Access* **8**, 72204–72215 (2020)
19. Nasir, N., et al.: Water quality classification using machine learning algorithms. *J. Water Process Eng.* **48**, 102920 (2022)
20. Ria, A., Cicalini, M., Manfredini, G., Catania, A., Piotto, M., Bruschi, P.: The SENSIPLUS: a single-chip fully programmable sensor interface. In: Saponara, S., De Gloria, A. (eds.) *ApplePies 2021*. LNEE, vol. 866, pp. 256–261. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-95498-7_36
21. Roitero, K., et al.: Detection of wastewater pollution through natural language generation with a low-cost sensing platform. *IEEE Access* **11**, 50272–50284 (2023)
22. Tripathi, M., Singal, S.K.: Use of principal component analysis for parameter selection for development of a novel water quality index: a case study of river ganga india. *Ecol. Ind.* **96**, 430–436 (2019)
23. Zhu, L., Husny, Z.J.B.M., Samsudin, N.A., Xu, H., Han, C.: Deep learning method for minimizing water pollution and air pollution in urban environment. *Urban Clim.* **49**, 101486 (2023)



Bandwise Attention in CycleGAN for Fructose Estimation from Hyperspectral Images

Divyani Tyagi^(✉) and Tushar Sandhan

Perception and Intelligence Lab, Electrical Department,
Indian Institute of Technology Kanpur, Kanpur, India
{divyanit23,sandhan}@iitk.ac.in

Abstract. We propose an innovative approach for generating Near-Infrared (NIR) hyperspectral images from Visible (VIS) hyperspectral imagery using our Bandwise Attention based CycleGAN(BA-CycleGAN). This framework introduces three key enhancements:(i) the integration of a Bandwise Attention mechanism within the architecture to appropriately attend to spectral bands in the hyperspectral images, (ii) the addition of a Spectral Angle Mapper(SAM) based consistency loss to preserve spectral characteristics crucial for hyperspectral imagery, (iii) replacing the convolution block with depthwise separable convolution block to significantly reduce the number of training parameters. The generated NIR images are subsequently concatenated with their corresponding VIS hyperspectral images, producing composite VIS-NIR hyperspectral datasets to improve fructose estimation through the classification of fruit sweetness levels. We conducted comprehensive experiments using three distinct datasets: VIS, NIR, and the combined VIS-NIR. Our findings demonstrate that the NIR images generated by BA-CycleGAN exhibit good spectral fidelity, closely mimicking the characteristics of actual NIR hyperspectral images. Moreover, the combined VIS-NIR dataset outperforms the individual VIS and NIR datasets in classifying fructose or sugar content levels in fruit.

Keywords: Spectral Angle Mapper · Hyperspectral images · CycleGAN · Depthwise Separable Convolutions · Bandwise Attention

1 Introduction

Traditional methods for fructose or sugar content estimation involve extracting juice from fruits and measuring soluble solids content (SSC) using refractometers [1]. The advent of hyperspectral imaging [2] in precision agriculture [3–5], has revolutionized the field of fruit quality assessment by offering a non-destructive and comprehensive solution. Hyperspectral imaging allows for the capture of detailed spectral information across a wide range of wavelengths, providing rich data on the biochemical composition, physiological status, and

structural characteristics of fruits. Hyperspectral imaging sensors frequently produce hundreds of narrow spectral bands from a single region on a fruit. Each pixel in the hyperspectral image can be thought of as a high-dimensional vector, with each entry representing the spectral reflectance at a certain wavelength. Among the spectrum ranges, Visible (VIS) wavelengths (400–700 nm) and Near-Infrared (NIR) wavelengths (700–2500 nm) have been considerably researched, although NIR hyperspectral imaging has gained importance due to its sensitivity to metabolic changes related to sugar content [6, 7]. NIR spectral signatures reflect the molecular vibrations of chemical bonds, offering insights into the internal composition of fruits and enabling the estimation of sugar content with high accuracy and non-destructively.

Despite the potential benefits of NIR hyperspectral imaging, its widespread adoption in fruit quality assessment is hindered by the limited availability of datasets and the high cost of hyperspectral cameras [8]. Acquiring NIR hyperspectral data for various fruit varieties and conditions is challenging and resource-intensive, often requiring specialized equipment and expertise. As an alternative approach, researchers have explored the spectrum reconstruction from conventional RGB images [8, 9]. While this method offers accessibility and practicality, it may not fully capture the spectral richness and specificity of hyperspectral imaging, limiting its effectiveness in accurate sugar content estimation.

In this study, we propose a novel approach to address the challenges associated with the limited availability of NIR hyperspectral data. Instead of relying on RGB images, which capture only three spectral bands (red, green, and blue), we leverage Visible (VIS) hyperspectral images, which offer spectral information across a broader range of wavelengths, for the generation of NIR hyperspectral data. VIS hyperspectral imaging provides more detailed spectral data, allowing for the capture of subtle variations in fruit composition and quality. Also, the VIS hyperspectral camera is relatively available at a lower cost than the NIR hyperspectral camera. To bridge the gap between the limited availability of NIR hyperspectral data and the availability of VIS hyperspectral data, we employ the improved Cycle Generative Adversarial Networks (CycleGANs) [10] to facilitate unpaired image-to-image translation. We have incorporated new architectural changes in CycleGAN and modified its loss function to make the generation adaptable for hyperspectral images. CycleGANs enable the generation of synthetic NIR hyperspectral images from VIS hyperspectral images, thereby enhancing the accessibility and practicality of NIR hyperspectral imaging for fruit quality assessment.

VIS wavelengths capture information about surface color, texture, and pigmentation, which can be related to their ripeness and indirectly to their sugar content, while NIR wavelengths can penetrate deeper into the fruit’s surface, providing information about internal structures and compositions, such as moisture content, sugar content, and other attributes relevant to fruit quality. Combining VIS and NIR hyperspectral imaging takes advantage of both methods’ strengths. This combination offers a comprehensive overview, potentially leading to more accurate estimations of sugar content. By combining both VIS and NIR, hyperspectral imaging captures a broader range of spectral features, leading to a more

comprehensive characterization of fruits. However, the cost of acquiring such a hyperspectral camera which captures from both ranges (400 nm to 1700 nm) is very high. So to resolve this issue, we concatenate VIS(400 nm to 900 nm) hyperspectral data with corresponding generated NIR(900 nm to 1700 nm) hyperspectral data and created a larger VIS-NIR(400 nm to 1700 nm) hyperspectral tensor as shown in Fig. 1. Note the proposed generative model will not see the optically unseen in the NIR band however it will learn the feature dynamics within VIS bands for a given fruit and then map it appropriately to NIR bands thereby amplifying the useful latent features.

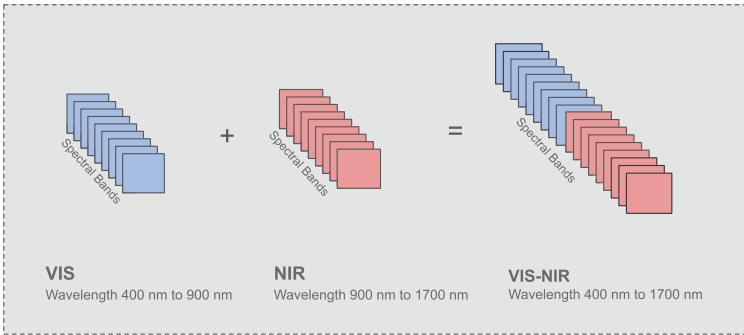


Fig. 1. Visible(VIS) spectral bands, near-infrared (NIR), and band-wise combination of VIS and NIR to get a composite VIS-NIR tensor.

The primary objective of this research is to leverage the complementary information contained within the VIS and NIR spectral regions to improve the classification of sweetness levels in fruits—a task of significant importance for agricultural quality control and consumer satisfaction. To this end, we conduct comprehensive experiments using three distinct datasets: VIS, NIR, and the combined VIS-NIR. We perform rigorous experimentation of the proposed method for assessing effectiveness in accurately predicting the sweetness level of various fruits.

This research not only highlights the potential of GANs in hyperspectral image synthesis but also paves the way for enhanced agricultural product classification through the innovative use of combined spectral data.

2 Related Works

2.1 Fructose or Sugar Content Estimation Methods

Chemical Methods for sugar content analysis, such as titration and refractometry, have been the gold standard for many years. Titration provides precise measurements [11]. Refractometry, on the other hand, measures the refractive index of a sugar solution, offering a quick and non-destructive means to assess sugar concentration [12].

Hardware-Based Methods consist of spectroscopic techniques, including Near-Infrared (NIR) [13] and Raman spectroscopy [14], which are prominent for their non-invasive nature and high accuracy. These methods analyze light absorption and scattering in fruit tissues, correlating specific wavelengths with sugar levels. Ultrasound is another non-destructive technique that estimates sugar content by measuring the acoustic properties of a fruit, which vary with sugar concentration [15].

Optical Methods include Infrared (IR) imaging [16] which are gaining popularity due to its ease of use and the ability to integrate with portable devices. IR imaging detects sugar content based on the distinct absorption and reflection patterns of IR light by sugars [17]. RGB color analysis, while less accurate, offers a cost-effective and rapid assessment of sugar content by analyzing the color changes in fruit skins, which correlate with ripeness and sugar levels but when accompanied with NIR spectroscopy, it gives better results [18].

2.2 Hyperspectral Imaging for Fruit Quality Assessment

Hyperspectral imaging has emerged as a promising alternative to traditional methods for fruit quality assessment [19–21]. By capturing detailed spectral information across a wide range of wavelengths, hyperspectral imaging enables non-invasive and comprehensive analysis of fruit composition and quality attributes. Near-Infrared (NIR) and Visible (VIS) spectral regions are used for various applications in fruit quality assessment, including sugar content detection. NIR hyperspectral imaging has gained prominence for its ability to detect sugar content in fruits based on the spectral signatures associated with biochemical changes related to sugar concentration [22]. Despite its potential, the widespread adoption of NIR hyperspectral imaging in fruit quality assessment is hindered by the limited availability of labeled datasets. Before hyperspectral images are given to Deep learning models, some preprocessing steps such as dimensionality reduction of hyperspectral images like Principal Component Analysis (PCA) [23] have also been used but crucial information in spectral bands gets lost and that’s how we came up with the idea of using a bandwise attention mechanism.

2.3 Generation of Hyperspectral Images

To address the challenges associated with the limited availability of NIR datasets, researchers have explored GANs for generating realistic spectral images to augment datasets, thereby addressing the challenge of limited data availability [24]. CycleGAN [10] architecture has been adapted for translating images between different spectral domains without paired training samples. This capability is particularly relevant for generating NIR images from VIS images, enabling the study of cross-spectral characteristics without the need for exact spectral matches. CycleGANs, in particular, have been employed to transform images such as translating RGB to single-band NIR images [25], single-band VIS images from single-band NIR image [26], RGB to RGB [27] transfer. As per our knowledge, we are the

first to do a multiband high-fidelity transformation for hyperspectral images. Our approach offers a data-driven solution to enhance the spectral resolution of imaging data and overcome the limitations of traditional hyperspectral imaging techniques.

For the evaluation of generated images, PSNR (Peak Signal to Noise Ratio) and SSIM (Structural Similarity) [28] objective evaluation indicators were used. Both these indicators are related to determining images' visual quality but hyperspectral images are more about spectral fidelity. So for qualitative analysis of generated hyperspectral images, we used Spectral Angle Mapper (SAM) distance [29].

Our study builds upon these foundational works by not only generating NIR hyperspectral images from VIS counterparts using BA-CycleGAN but also by demonstrating the utility of combined VIS-NIR hyperspectral images for the classification of fruit sweetness levels.

3 Our Method

3.1 Preliminaries

CycleGAN [10] is designed for unpaired image-to-image translation tasks, where the network learns to map images from one domain to another without the need for paired training data. CycleGAN consists of two main components: a generator and a discriminator.

Generator in CycleGAN aims to translate images from one domain to another. It consists of two sub-networks: a forward generator $G : X \rightarrow Y$ that maps images from domain X to domain Y , and a backward generator $F : Y \rightarrow X$ that maps images from domain Y back to domain X . The generators aim to minimize the cycle consistency loss, ensuring that the translated image can be reconstructed back to the original domain.

Discriminator in CycleGAN aims to distinguish between translated images and real images from the target domain. It consists of two discriminators D_X and D_Y for domains X and Y respectively. The discriminators aim to differentiate between real and fake images, while the generators aim to fool the discriminators.

3.2 Proposed Method

We have introduced new architectural changes in the original CycleGAN by incorporating a bandwise attention mechanism and new losses. All these changes are made so that the proposed GAN architecture can handle the unique characteristics of hyperspectral images, such as its high dimensionality and spectral information, and also extract both spectral and spatial features.

Network Architecture. The architecture of BA-CycleGAN is shown in Fig. 2. Following architectural changes have been made in the original CycleGAN [10].

First Layer Modification: We have modified the input layers to accommodate the depth of hyperspectral images. The first convolution layer is adjusted to accept

L input channels instead of three for RGB images. This involves changing the filter shape in the first layer from [filter-height, filter-width, 3, num-filters] to [filter-height, filter-width, L , num-filters].

Bandwise Attention in the Encoder of Generator: We have introduced a bandwise attention mechanism at the beginning of the generator of the CycleGAN. The hyperspectral input image, with its L spectral bands, is fed into the generator. This input immediately passes through the bandwise attention mechanism. The mechanism calculates attention scores for each of the L spectral bands. These attention scores allow the network to emphasize more important bands and de-emphasize less relevant ones. The attended feature map is fed into the encoding block. The main objective of integrating this mechanism is to allow the model to learn intricate dependencies and relationships between different spectral bands, leveraging the rich spectral information.

Bandwise Attention in the Decoder of Generator: A second instance of the Bandwise Attention mechanism is placed before the decoding block. It allows the network to refine its focus on spectral relationships after processing through the encoder and transformer blocks. It gives the decoder a chance to emphasize the most relevant spectral features just before final image reconstruction.

Depthwise Separable Convolution: The first layer’s parameters increased significantly in both the discriminator and generator structure due to more input channels which leads to the overall increase in trained parameters. We have replaced convolutional layers with depthwise separable convolution [30] because of its efficiency in processing high-dimensional hyperspectral data with fewer parameters and computational resources compared to standard convolution. It also facilitates effective learning of spectral-spatial features by processing spatial information within each spectral band independently before integrating across the spectrum.

Bandwise Attention Mechanism. is an attention mechanism where we determine how much each band should be attended. Note that this is different from the normal attention mechanism in the way that here we are calculating the attention score for each band rather than each pixel. We performed the below steps to calculate the attention score.

Data Representation: We represented the hyperspectral image as a 3D tensor X of shape $[n, n, L]$, where (n, n) is spatial resolution and L is the number of spectral bands. We can consider each pixel as a vector of L spectral values corresponding to the L bands at that pixel.

Flatten for Processing: We converted X from $[n, n, L]$ to $[N, L]$, where $N = n \times n$ is the total number of pixels.

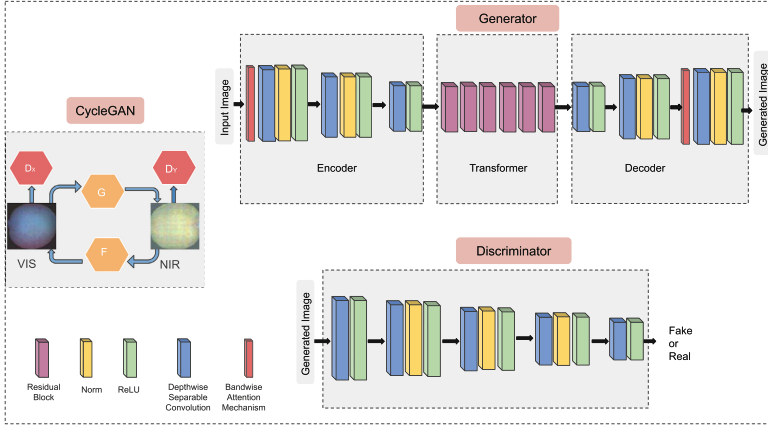


Fig. 2. Architecture of BA-CycleGAN for hyperspectral images. Our proposed Generator includes a Bandwise Attention Mechanism and convolution layers are replaced by Depthwise separable convolution.

Calculation of Queries (Q), Keys (K), and Values(V): We computed the Q , K , and V matrices as given in Eq. 1, 2, and 3 respectively.

$$Q = W^Q X \tag{1}$$

$$K = W^K X \tag{2}$$

$$V = W^V X \tag{3}$$

Here, W^Q , W^K , and W^V are weight matrices of dimensions $[N, N]$.

Compute Attention Scores: We calculate the dot product between queries and keys, then scale by \sqrt{N} to facilitate gradient stability:

$$A = \text{softmax} \left(\frac{QK^T}{\sqrt{N}} \right) \tag{4}$$

A is of size $[L, L]$, where each entry a_{ij} represents the attention weight of band i to band j .

Output Computation: We multiplied the attention matrix by the values to obtain the output:

$$O = AV \tag{5}$$

This results in a matrix O of shape $[N, L]$, where each row now contains a weighted combination of features across all bands based on their spectral information and spatial relationships.

Reshape to Original Spatial Dimensions: We reshaped output from $[N, L]$ back to $[n, n, L]$ to align with the original spatial structure of the image. This reshaped output was used as input to further layers.

Loss Function: The generator aims to minimize the following loss functions:

$$\mathcal{L}_{GAN}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log(1 - D_Y(G(x)))] \quad (6)$$

$$\mathcal{L}_{GAN}(F, D_X, Y, X) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D_X(x)] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\log(1 - D_X(F(y)))] \quad (7)$$

$$\mathcal{L}_{\text{cycle}}(G, F) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1] \quad (8)$$

However, we have introduced a new loss which is SAM-based spectral consistency loss. SAM calculates the cosine of the angle between the spectral signature vectors of the transformed NIR hyperspectral image and the original NIR hyperspectral image. It ensures that the transformed NIR images maintain a spectral profile that is consistent with true NIR images by measuring how well the translated image preserves the spectral directionality, regardless of the magnitude, making it less sensitive to illumination differences. Spectral Angle Mapper (SAM) between two spectral vectors \mathbf{A}_n and \mathbf{B}_n in an L -dimensional space (where L is the number of spectral bands) is calculated for each pixel and then it is integrated over all the pixels in the hyperspectral image as shown in the equation below:

$$\mathcal{L}_{SAM} = \frac{1}{N} \sum_{n=1}^N \arccos \left(\frac{\mathbf{A}_n \cdot \mathbf{B}_n}{\|\mathbf{A}_n\|_2 \|\mathbf{B}_n\|_2} \right) \quad (9)$$

Here, N represents the number of pixels, \mathbf{A}_n and \mathbf{B}_n denote the L -dimensional spectral vectors at the n -th pixel for the translated and real images, respectively.

The total loss function, \mathcal{L} , for the BA-CycleGAN incorporating SAM-based spectral consistency is defined as:

$$\mathcal{L} = \mathcal{L}_{GAN} + \lambda_{cyc} \mathcal{L}_{cyc} + \lambda_{SAM} \mathcal{L}_{SAM} \quad (10)$$

where \mathcal{L}_{GAN} represents the adversarial loss, \mathcal{L}_{cyc} is the cycle consistency loss, \mathcal{L}_{SAM} is the Spectral Angle Mapper (SAM) based spectral consistency loss. And λ_{cyc} , and λ_{SAM} are hyper-parameters controlling the relative importance of each loss component.

4 Experiment

4.1 Dataset

The primary goal is to examine the fructose or sugar content level of the fruit. Two datasets have been utilized for validating our model. The first one is the DeepHS [31] dataset consisting of VIS and NIR hyperspectral images of Kiwi

fruit. Each hyperspectral recording from VIS dataset encompasses 224 spectral bands within the visible range of the electromagnetic spectrum, spanning from 400 nm to 900 nm while in the case of NIR dataset, 252 bands are lying from 900 nm to 1700 nm.

The second dataset is Wax Apple Hyperspectral-Image Open Dataset (WA-HSI) [32]. In this, 136 wax apple samples are separated into slices (1034 slices in total) for hyperspectral image data collection. These raw hyperspectral images have 1367 bands from 400 to 1700 nm. To validate the model on this dataset, each small cube (size $20 \times 20 \times 1367$) of HSIs datasets was separated into two cubes with different band ranges, including 400–900 nm (size $20 \times 20 \times 943$) and 900–1700 nm (size $20 \times 20 \times 424$). The numbers 943 and 424 represent the individual bands from 400–1000 nm and 900–1700 nm. In both datasets, each recording has Brix values for sugar content analysis. We are going to classify hyperspectral images into 3 classes (less sweet, perfect, and very sweet) based on Brix levels which are correlated with the fructose or sugar content level in the fruit.

4.2 Results

For qualitative analysis using the DeepHS [31] dataset, some samples of generated NIR images from different GANs after training for 120 epochs are visualized as shown in Fig. 3. The images are just the RGB representation of hyperspectral images, although BA-CycleGAN generated images look visually better than CycleGAN, but their visual inspection will not be very fruitful here, so we focused on quantitative analysis by calculating the SAM distance [29] between generated NIR images and original NIR images as shown in Table 1, and classification accuracy for fructose or sugar content estimation.

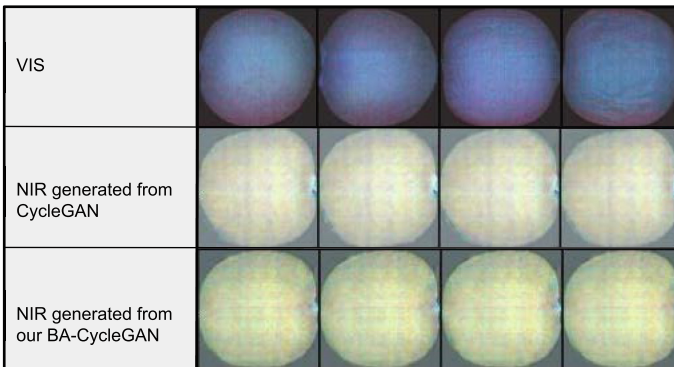


Fig. 3. Sample of generated NIR images from BA-cycleGAN and original cycleGAN [10]

We combined the generated dataset with the original dataset to test the accuracy of sugar content classification for the VIS, NIR, and VIS-NIR hyperspectral

Table 1. Comparison of SAM distance calculated between generated NIR images and original NIR images.

Model	Dataset	SAM Distance in degrees (Lower is better)
CycleGAN	DeepHS	28.2
	WA-HSI	21
BA-CycleGAN	DeepHS	14.0
	WA-HSI	7

as shown in Table 2, 3, and 4 respectively. A comparison with the original dataset and original dataset + data generated from CycleGAN has also been made to check the reliability of the generated dataset for classification. We have compared the classification results using DeepHS [31], ResNet 50 [30], VGG 16 [33], DenseNet 121 [34], and EfficientNet B7 [35]. In all these models, the first layer has been modified to adapt large dimensions of hyperspectral images.

Table 2. Test accuracy(%) for different models for fructose or sugar content when trained on VIS hyperspectral datasets.

Model	Dataset	Original data	Original data + data from CycleGAN	Original data + data from BA-CycleGAN
ResNet 50 [30]	DeepHS	52	54	60
	WA-HSI	61	62	62
VGG 16 [33]	DeepHS	50	52	55
	WA-HSI	55	58	60
DeepHS [31]	DeepHS	50	52	55
	WA-HSI	61	62	62
DenseNet 121 [34]	DeepHS	54	54	58
	WA-HSI	62	64	64
EfficientNet B7 [35]	DeepHS	50	52	60
	WA-HSI	61	62	62

5 Ablation Studies

We study the impact of each mechanism we introduced in CycleGAN. All the ablation studies are performed on the DeepHS dataset.

Table 3. Test accuracy(%) for different models for fructose or sugar content when trained on NIR hyperspectral datasets.

Model	Dataset	Original data	Original data + data from CycleGAN	Original data + data from BA-CycleGAN
ResNet 50 [30]	DeepHS	50	66	70
	WA-HSI	62	63	63
VGG 16 [33]	DeepHS	40	58	60
	WA-HSI	55	59	60
DeepHS [31]	DeepHS	30	58	60
	WA-HSI	58	58	60
DenseNet 121 [34]	DeepHS	50	66	70
	WA-HSI	61	62	63
EfficientNet B7 [35]	DeepHS	50	66	72
	WA-HSI	62	64	64

Table 4. Test accuracy(%) for different models for fructose or sugar content when trained on VIS-NIR hyperspectral datasets.

Model	Dataset	Original data + data from CycleGAN	Original data + data from BA-CycleGAN
ResNet 50 [30]	DeepHS	70	77
	WA-HSI	71	74
VGG 16 [33]	DeepHS	60	67
	WA-HSI	77	78
DeepHS [31]	DeepHS	60	67
	WA-HSI	79	80
DenseNet 121 [34]	DeepHS	72	68
	WA-HSI	76	80
EfficientNet B7 [35]	DeepHS	72	73
	WA-HSI	79	80

5.1 Effect of Using SAM-Based Consistency Loss

We assessed the impact of incorporating SAM-based consistency loss into the BA-CycleGAN framework using the DeepHS dataset. This investigation involved comparing two configurations: BA-CycleGAN with, and without SAM-based consistency loss. The introduction of the SAM-based loss significantly improved the spectral fidelity of generated NIR images, as evidenced by lower average SAM values in Table 5, suggesting a closer spectral match with real NIR images. Moreover, the augmented model facilitated superior classification accuracy in determining fruit sweetness levels, particularly when leveraging combined VIS-NIR datasets.

Table 5. Comparison of SAM distance calculated between generated NIR images and original NIR images when SAM-based consistency loss is removed on.

Model	SAM Distance in degrees
BA-CycleGAN without SAM loss	22.7
BA-CycleGAN with SAM loss	14

5.2 Effect of Using Bandwise Attention Mechanism

Bandwise attention mechanism learns better mapping of bands in hyperspectral images and thus results in better correlated images as shown in Table 6.

Table 6. Comparison of SAM distance calculated between generated NIR images from BA-CycleGAN and original NIR images when bandwise attention mechanism is removed.

Model	SAM Distance in degrees
without Bandwise Attention Mechanism	24
with Bandwise Attention Mechanism	14

5.3 Effect of Using Depthwise Separable Convolution

As hyperspectral images have very high dimensionality, incorporating them with CycleGAN results in a very large number of training parameters and thus large training time. For an input image of size 224*60*60 (from DeepHS dataset), we have obtained the training parameters for each layer in the discriminator and generator architecture. Tables 7 and 8, we compare the training parameters for each layer, when replacing convolution layers with depthwise separable convolution layers. The depthwise separable convolutions have reduced the parameter count by about 89 % and 93% in generator and discriminator architecture respectively (Table 9).

Table 7. Comparison of training parameters of discriminator architecture with or without depthwise convolution layers.

Generator Layers	Training parameters without depthwise separable convolution	Training parameters with depthwise separable convolution
Encoder layer 1	129,088	16,640
Encoder layer 2	73,856	8,960
Encoder layer 3	295,168	34,304
Transformer Block	10,621,440	1,223,424
Decoder layer 1	295,040	35,456
Decoder layer 2	73,792	9,536
Decoder layer 3	258,272	15,200
Total	11,746,656	1,343,520

Table 8. Comparison of training parameters of generator architecture with or without depthwise convolution layers.

Discriminator Layers	Training parameters without depthwise separable convolution	Training parameters with depthwise separable convolution
layer 1	229,440	18,208
layer 2	131,200	9,408
layer 3	524,544	35,200
layer 4	2,097,664	135,936
layer 5	8,193	9,217
Total	2,991,041	207,969

Table 9. Comparison of training time of BA-CycleGAN with or without depthwise convolution layers.

Model	Training time for 120 epochs
without Depthwise Convolution layers	68 h
with Depthwise Convolution layers	52 h

6 Conclusion

Our study advances the generation of NIR hyperspectral images from VIS data through BA-CycleGAN, highlighting the crucial role of spectral fidelity in hyperspectral imaging. By incorporating a SAM-based consistency loss and a bandwise spectral attention block, we not only enhanced image quality but also significantly improved the classification of fruit fructose levels. We have compared the classification results for VIS, NIR, and VIS-NIR datasets and found that VIS-NIR perform better than NIR dataset. This research is not limited to fructose estimation it can be further used for water and other nutrient estimation not only in fruits but vegetables and leaves also. VIS-NIR bands can be used extensively in many domains, and so the method can also be extended in areas other than agriculture. Future efforts could further optimize these modifications, broadening the potential applications of this technology in environmental monitoring, mineralogy, and beyond, making advanced hyperspectral imaging more accessible and impactful across diverse research fields.

Acknowledgements. This work is supported by IIT Kharagpur AI4ICPS I Hub Foundation, a.k.a AI4ICPS under the aegis of DST.

References

1. Magwaza, L.S., Opara, U.L.: Analytical methods for determination of sugars and sweetness of horticultural products—A review. *Scientia Horticulturae* **184**, 179–192 (2015)

2. Cullen, P.J., Downey, G., Frias, J.M., Gowen, A.A., O'Donnell, C.P.: Hyperspectral imaging - an emerging process analytical tool for food quality and safety control. *Trends Food Sci. Technol.* **18**, 590–598 (2007)
3. Amigo, J.M., Babamoradi, H., Elcoroaristizabal, S.: Hyperspectral image analysis: a tutorial. *Analytica chimica acta* **896**, 34–51 (2015)
4. Pádua, J., et al.: Hyperspectral imaging: a review on uav-based sensors, data processing and applications for agriculture and forestry. *Remote Sens.* **9**, 1110 (2017)
5. Chu, B., Fan, Y., Zhu, H., et al.: Hyperspectral imaging for predicting the internal quality of kiwifruits based on variable selection algorithms and chemometric models. *Sci. Rep.* **7**, 7845 (2017)
6. Chen, N.Z.L.Z.X.Z.J., Bai, T.: Hyperspectral detection of sugar content for sugar-sweetened apples based on sample grouping and spa feature selecting methods. *Infrared Phys. Technol.* **125**, 104240 (2022)
7. Ma, T., Li, X., Inagaki, T., Yang, H., Tsuchikawa, S.: Noncontact evaluation of soluble solids content in apples by near-infrared hyperspectral imaging. *J. Food Eng.* **224**, 53–61 (2018)
8. Oh, S.W., Brown, M.S., Pollefeys, M., Kim, S.J.: Do it yourself hyperspectral imaging with everyday digital cameras (2016)
9. Su, R., Fu, Q., Zhang, J., et al.: A survey on computational spectral reconstruction methods from rgb to hyperspectral imaging. *Sci. Rep.* **12**, 11905 (2022)
10. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks, pp. 2242–2251 (2017)
11. Islam, M.K., Khan, M.Z.H., Sarkar, M.A.R., Absar, N., Sarkar, S.K.: Changes in acidity, TSS, and sugar content at different storage periods of the postharvest mango (*Mangifera indica* L.) influenced by Bavistin DF. *Int. J. Food Sci.* **2013**(1), 939385 (2013)
12. Kataoka, I., Tomana, T., Sugiura, A.: Use of refractometer to determine soluble solids of astringent fruits of Japanese persimmon (*Diospyros Kaki* L.). *J. Horticult. Sci.* **58**, 241–246 (1983)
13. Rambla, F.J., Garrigues, S., De La Guardia, M.: PLS-NIR determination of total sugar, glucose, fructose and sucrose in aqueous solutions of fruit juices. *Analytica Chimica Acta* **344**(1–2), 41–53 (1997)
14. Castro-Ramos, J., Cerecedo-Núñez, H.H., González-Viveros, N., Gómez-Gil, P.: On the estimation of sugars concentrations using raman spectroscopy and artificial neural networks. *Food Chem.* **352**, 129375 (2021)
15. Mizrach, A.: Ultrasonic technology for quality evaluation of fresh fruit and vegetables in pre- and postharvest processes. *Postharvest Biol. Technol.* **48**, 315–330 (2008)
16. Sugiyama, J.: Visualization of sugar content in the flesh of a melon by near-infrared imaging. *J. Agric. Food Chem.* **47**, 2715–2718 (1999)
17. Guyer, D.E., Rady, A.M.: Evaluation of sugar content in potatoes using nir reflectance and wavelength selection techniques. *Postharvest Biol. Technol.* **103**, 17–26 (2015)
18. Moltó, E., Blasco, J., Steinmetz, V., Roger, J.M.: On-line fusion of colour camera and spectrophotometer for sugar content prediction of apples. *J. Agric. Eng. Res.* **73**, 207–216 (1999)
19. Wang, B., et al.: The applications of hyperspectral imaging technology for agricultural products quality analysis: a review. *Food Rev. Int.* **39**(2), 1043–1062 (2023)
20. Gao, Z., Shao, Y., Xuan, G., Wang, Y., Liu, Y., Han, X.: Real-time hyperspectral imaging for the in-field estimation of strawberry ripeness with deep learning. *Artif. Intell. Agric.* **4**, 31–38 (2020)

21. Shang, J., Tan, T., Feng, S., et al.: Nondestructive quality assessment and maturity classification of loquats based on hyperspectral imaging. *Sci. Rep.* **13**, 13189 (2023)
22. Wei, S.Z.D.Y.X., He, J.: Modeling for ssc and firmness detection of persimmon based on nir hyperspectral imaging by sample partitioning and variables selection. *Infrared Phys. Technol.* **105**, 103099 (2020)
23. Rodarmel, C., Shan, J.: Principal component analysis for hyperspectral image classification. *Survey. Land Inf. Sci.* **62**(2), 115–122 (2002)
24. Wambugu, N., et al.: Hyperspectral image classification on insufficient-sample and feature learning using deep neural networks: a review. *Int. J. Appl. Earth Observat. Geoinf.* **105**, 102603 (2021)
25. Dou, H., Chen, C., Hu, X., Peng, S.: Asymmetric cyclegan for unpaired nir-to-rgb face image translation, pp. 1757–1761 (2019)
26. Wang, H., Zhang, H., Yu, L., Yang, X.: Facial feature embedded CycleGAN for VIS–NIR translation. *Multidimen. Syst. Signal Process.* **34**(2), 423–446 (2023)
27. Yasarla, R., Sindagi, V.A., Patel, N.M.: Unsupervised restoration of weather-affected images using deep gaussian process-based cyclegan, pp. 1967–1974 (2022)
28. Ziou, D., Horé, A.: Image quality metrics: PSNR VS. SSIM, pp. 2366–2369 (2010)
29. Paikaray, B.K., Mishra, R., Dash, S., Chakravarty, S.: Hyperspectral image classification using spectral angle mapper, pp. 87–90 (2021)
30. Gomez, A.N., Kaiser, L., Chollet, F.: Depthwise separable convolutions for neural machine translation (2017)
31. Varga, J.M., Amadeus, L., Zell, A.: Measuring the ripeness of fruit with hyperspectral imaging and deep learning. In: *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8 (2021)
32. Yan, Y.J., Huang, C.C., Chen, C.J., et al.: Sugariness prediction of syzygium samarangense using convolutional learning of hyperspectral images. *Sci. Rep.* **12**, 2774 (2022)
33. Zhu, L., Chen, Y., Ghamisi, P., Benediktsson, J.A.: Generative adversarial networks for hyperspectral image classification. *IEEE Trans. Geosci. Remote Sens.* **56**(9), 5046–5063 (2018)
34. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. *coRR* (2018)
35. Tan, M., Le, Q.V.: Efficientnet: rethinking model scaling for convolutional neural networks (2020)



Unsupervised Multi-level Search and Correspondence for Generic Voice-Face Feature Spaces

Jing Sun and Jianbo Su(✉)

Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China
sunjing4231@sjtu.edu.cn

Abstract. This paper focuses on the challenge of searching a shared feature space for face and voice modalities. Studies of human intelligence have shown that people can link faces and voices. However, computational intelligence research has paid less attention to the relationship between voice and face. The objective of this work is to identify generic features that are generalized to both visual and audio modalities, enabling the matching of faces and voices. To achieve a better approximation of the feature spaces of two modalities, a multi-level alignment approach is applied to their features. For individual sample pairs, a contrast learning approach is exploited. For the overall feature distribution, an optimal transport method is used. The impact of pre-trained weights on this task is explored. Compared to the simple contrastive learning method, inclusion of overall feature alignment improves both verification and matching accuracy. Extensive experimental results suggest that the overall distribution alignment is useful for the cross-model feature matching task. Also, the use of pre-trained parameters can improve the results in certain situations.

Keywords: Voice-face association · Cross-modal embeddings · Optimal transport

1 Introduction

The mapping between voice and face is a crucial aspect of human intelligence. Therefore, studying the feature correspondence between voice and face is an essential research component of computational intelligence. The ability to associate voice and face has potential applications in areas such as criminal investigation and intelligent dubbing, where the extension of human identity features is required.

This paper addresses the question of whether machine learning can match the features of data from different modalities, specifically, faces and voices. Face and voice cues are among the most commonly used non-invasive and easily accessible cues in various identification tasks [1]. Studies in cognitive science and

neuroscience have shown that humans tend to integrate audiovisual information when performing various perceptual tasks [2]. It has also been confirmed that humans can accurately match unfamiliar facial images with corresponding voice recordings and vice versa [3, 4].

Considerable research has been conducted on the problem of cross-modal features. These studies use the feature of one modality to enhance detection [5, 6] or generate data [7, 8] in the other modality. The ability to combine information from different modalities can improve detection results and is an important way to bring machine intelligence closer to the way humans process information. The interaction between information from different modalities is also very important for understanding the world. For example, videos can have different meanings with and without background sounds [9]. Methods such as cross-domain attention have focused on how to extract more effective features in these related problems. In this paper, we focus on how to measure the similarity of two feature spaces and, through the design of loss functions, find a generalised space for features from different modalities.

Face-voice correlation is a cross-modal problem that has also been investigated. Some studies [5, 10, 11] use contrast learning to bring the corresponding cross-modal features of the samples closer together, while pushing the features of different people further apart. In addition, some researchers [12, 13] propose to implicitly limit the differences between the two distributions by attaching a classifier to the feature extraction network. Wen et al. [12] suggest that using supervision from other information can eliminate differences between different modal features, and Nagrani et al. [13] approach the audio-face matching task as a multi-classification problem. An alternative method proposed by Cheng et al. [14] is to use adversarial learning. They introduce a novel adversarial deep semantic matching network that uses a discriminator to bridge the gap between voice and face features while maintaining semantic consistency.

However, these methods do not consider the alignment of the overall distribution, which we believe is crucial for this task. In this paper, both single-sample and overall alignment is applied to achieve the matching between facial and vocal modalities. Different from the implicit distribution constraint in existing methods, Optimal Transport (OT) is used to directly align the overall feature distributions. Furthermore, to make use of the supplementary information, pre-trained parameters are employed during the training process, and the effect of these parameters is analysed.

The remainder of this paper is structured as follows: Sect. 2 provides an introduction to related studies. Section 3 describes the methods used in the paper, including single sample pair feature alignment, overall distribution alignment, and combination of the two. Specific experimental details are presented in Sect. 4. In Sect. 5 we analyse and discuss the experimental results, followed by conclusions in Sect. 6.

2 Related Work

This paper focuses on establishing a correlation between two modal features to obtain a generalized feature. Previous studies have identified three types of methods for feature alignment. The first involves directly pulling together features of the same person using contrastive methods [5, 10, 11]. The second involves using classifiers attached to the feature extraction network to implicitly achieve a limit [12, 13]. Finally, adversarial methods can eliminate differences between features of several modalities by adding a classifier [15].

Feature extraction is a crucial aspect of this task. As various modal features possess distinct characteristics, the feature extraction network typically varies across modalities [16–18]. A single stream network can be used for feature extraction after specific data processing [19]. In this work, the encoder networks are selected based on the properties of different modal data to extract more representative features.

The optimal transport method has been used in several situations. It has various applications, such as image processing, machine learning, statistics, and computational fluid dynamics [20, 21]. In these studies, data from different datasets are considered as separate domains. Data from one domain is then transferred to another to achieve good recognition results on both domains. Previous study [22] has showed that optimal transport is an effective way to realize domain adaptation and the convergence is regarded.

In [23] optimal transport is applied to realise cross-domain speaker verification, which is close to this study. Because they both focus on the way to extract shared human features through different field of data. In this study, the concept of ‘different domain’ is extended to ‘different modality’. Optimal transport is used to reduce the distance between the feature spaces of different modalities, thereby achieving domain alignment in this cross-modal problem.

Deep migration learning algorithms based on domain adaptive theory typically rely on pre-trained models [24, 25]. The function space of deep networks is very large, and the pre-training process can effectively reduce the allowable function space, thus greatly reducing the generalisation error on the target domain. This study borrows the idea to better extract features from different modality. In comparison to the existing large-scale face dataset, which contains over 9,000 individuals [26, 27], the current voice-face corresponding dataset is relatively small, with just over 1,200 individuals [28]. The utilization of pre-trained parameters incorporates information from other extensive datasets, enabling the features to more effectively represent the original data.

3 Cross-Modal Feature Alignment Task

Deep learning methods involve encoding the original data using deep neural networks to obtain high-dimensional features. Existing research suggests a correlation between an individual’s facial and vocal characteristics, suggesting the potential to align the feature space of the two modalities. The available dataset

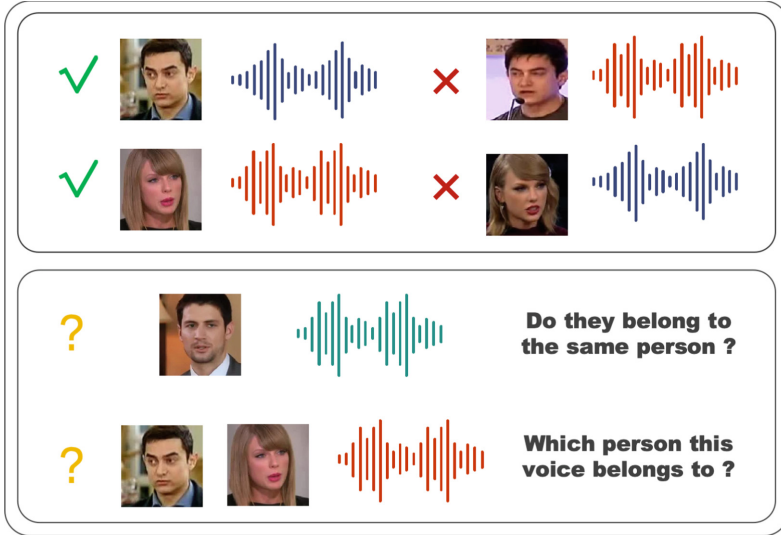


Fig. 1. The task presented in this paper. The top diagram displays the corresponding image and sound, while the bottom diagram represents the problem to be solved. The model learns the shared feature space between the two modalities and eventually determine whether a pair of samples belongs to the same person, or which of the two images (sounds) corresponds to the given sound (image).

comprises pairs of facial images and sounds. These pairs were obtained by capturing videos from the web and extracting image frames at regular intervals. By training a deep neural network on ‘true’ voice-face pairs and restricting the features in the two modes to be similar, the high-level features of the two modes share a common space. Then it can be determined whether a given pair of face and voice belongs to the same person or which face from a set corresponds to a particular voice. Figure 1 is a schematic representation of this problem.

The method utilises two distinct neural networks to extract facial and vocal features separately. Subsequently, feature alignment is performed. Each facial image x_f and voice signal x_v are encoded separately into features by neural networks: $e_f = f(x_f)$, $e_v = g(x_v)$, $f(\cdot)$ and $g(\cdot)$ are deep neural networks used to extract features from images or audio. To achieve cross-modal feature matching, the distribution of e_f and e_v should be close to each other.

4 Multi-level Alignment of Voice and Face Features

To efficiently extract features from different modalities, two independent networks are used to encode input data. Each network is responsible for extracting features for a specific modality, and the output of each encoder is guaranteed to have the same feature dimension. After extracting the features, methods are

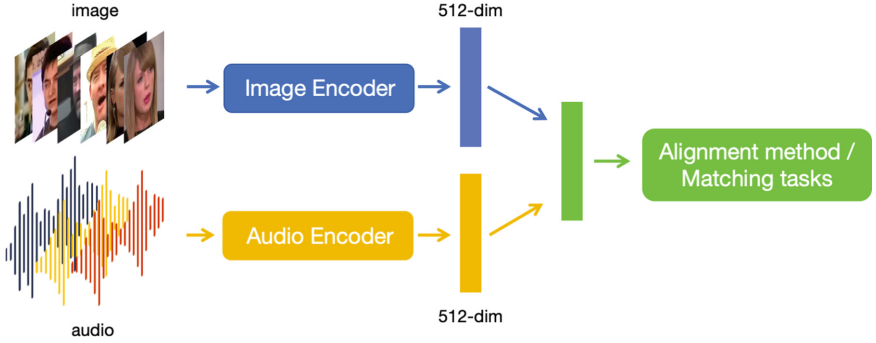


Fig. 2. Representation of the overall structure. The image and audio signals are initially encoded using different encoders to obtain features of the same dimension. These features are then aligned using the methods described in Sect. 3. The aligned features are used for subsequent tasks where their similarity is compared.

implemented for feature alignment to establish correspondence between different modalities. The overall network structure is shown in Fig. 2.

4.1 Single-Sample Feature Alignment Using Contrastive Method

A very intuitive idea to realize the correspondence of features between two modalities is to use a comparative learning approach. For the features (f_1, v_1) , (f_2, v_2) , \dots , (f_n, v_n) of different individuals on two modalities, it is necessary to reduce the distance between (f_i, v_i) and increase the distance between $(f_i, v_j), i \neq j$ at the same time, and this can be achieved by designing the following loss function:

$$L_c(v_i, f_i) = -\log \frac{\exp(v_i^T f_i / \tau)}{\exp(v_i^T f_i / \tau) + \sum_{f_j \in N_i^F} \exp(v_i^T f_j / \tau) + \sum_{v_j \in N_i^V} \exp(v_j^T f_i / \tau)}, \quad (1)$$

In each epoch with size N , there are N pairs of positive samples and $2N(N-1)$ pairs of negative samples. For a pair of positive sample (v_i, f_i) , each feature can product $(N-1)$ pair of negative samples: $(v_i, f_j)|_{(f_j \in N^F)}$ and $(v_j, f_i)|_{(v_j \in N^V)}$, thus there are $2(N-1)$ negative distances, which need to be pull further.

This method only considers the direct correspondence of the sample features of the two modalities and does not take into consideration the alignment of the samples to the overall distribution. This can lead to slow convergence of the model and the possibility that (f_i, v_i) may be similar even when $i \neq j$. If this happens, using the above training method will lead to poorer model results. Therefore, it is necessary to include the overall distribution alignment method.

4.2 Overall Distribution Alignment Using Optimal Transport

Optimal Transport (OT) aims to find a global optimal transport plan (or coupling) to convert one probability distribution shape into another shape with the least effort [29]. It focuses on efficiently transferring data between two distributions while maintaining a constant level of quality. OT defines an effective geometry-aware Wasserstein distance and it preserves the shapes of probability distributions during adaptation.

To achieve optimal transmission, it is necessary to solve a mathematical problem known as the ‘cost optimization problem’. The objective is to identify an optimal transmission strategy that minimises the cost of transmitting data between two given distributions. In each epoch, the distance of different features: (f_a, v_b) is measured by Euclidean distance:

$$D(f_a, v_b) = \|f_a - v_b\|^2. \quad (2)$$

γ is solved to minimize:

$$L_{OT}^* = \sum_{a,b} D(f_a, v_b) \gamma(f_a, v_b). \quad (3)$$

The difference of distributions between two modalities can be measured by the following loss function:

$$L_{OT}(f, v) = \min_{\gamma \in \Pi(f, v)} \sum_{a,b} D(f_a, v_b) \gamma(f_a, v_b). \quad (4)$$

L is the distance between two features, γ is the transport plan (or coupling) between the domains of two modalities.

4.3 Multi-level Feature Alignment

Both alignment methods are applied during the training phase. When considering only L_c , the symmetry in the overall distribution is neglected. Similarly, when only L_{OT} is considered, the direct sample correspondence information is missing. Therefore, these two kinds of losses need to be utilized simultaneously in the network training process. A parameter α is used to adjust the weights of the two parameters and the final loss function is:

$$L = L_c + \alpha * L_{ot}. \quad (5)$$

In this way, both the direct matching of the feature of sample pairs and the alignment of two modalities features distributions are used. During training, α needs to be dynamically tuned to balance the effects of the two loss functions (Fig. 3).

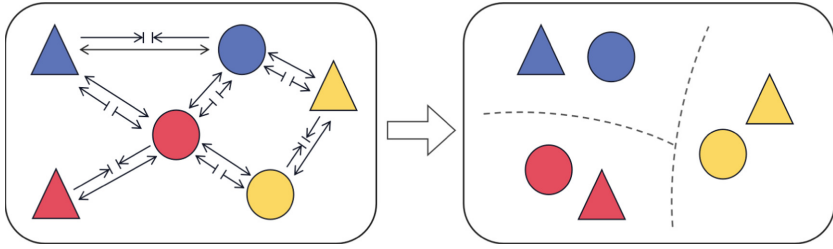


Fig. 3. A schematic representation of feature matching. Different colours are used to distinguish between individuals, while different shapes represent different modalities. After training, the features of different modalities of the same individual should be similar, whereas there should be noticeable differences between the features of different individuals.

4.4 Verification or Matching Based on Feature Similarity

Verification or matching tasks are performed by comparing the similarity of these features. During feature matching, specific similarity metrics are used to measure the degree of similarity between face images and speech segments. These metrics, such as Euclidean distance, cosine similarity, and correlation coefficient, quantify the similarity between two features and aid in determining whether they belong to the same person. A threshold is set to determine whether two features are considered to belong to the same person. The optimal threshold should be determined based on historical data and practical application scenarios. Cross-validation can be used to find a suitable threshold.

5 Preparation for Feature Extraction and Alignment

5.1 Dataset

The VoxCeleb [28] dataset was used in our experiments. It comprises 21,063 video clips of 1251 celebrities, extracted from videos uploaded to YouTube. The dataset is gender-balanced and includes speakers of different ethnicities, accents, professions, and ages. The face images are generated by cropping the video frames using facial bounding boxes. The voice clips are extracted from the video soundtracks with a 16kHz sampling rate at 16-bit PCM.

The test set is annotated with various demographic attributes to create five testing groups: the unstratified group (U), gender stratified group (G), nationality stratified group (N), age stratified group (A), and gender and nationality stratified group (GN). Table 1 displays the composition of each subgroup. In the experiments, each letter corresponds to a group where the individuals to be discriminated are from the same classification. For instance, in a group G experiment, the images and voices that need to be judged as belonging to the same person are either all from males or all from females. Matching and verification tasks are conducted under different demographic partitions.

Table 1. Composition of the dataset. The letters in brackets correspond to the groupings in the following table of experimental results.

Class	Gender(G)		Ethnicity(E)						Age group(A)							
	m.	f.	1	2	3	4	5	6	<19	20 s	30 s	40 s	50 s	60 s	70 s	≥80
Number	150	100	1	10	19	13	189	182	27	77	58	43	21	14	8	

The original face image is used as input for the image part. Each image is cropped to $122 * 122$ and randomly flipped horizontally. The sound part uses 80-dimensional MFCCs as input features. These are extracted from a 25-millisecond window with a frame shift of 10 milliseconds for a 2-s randomly cropped segment of each audio clip. SpecAugment [30] is applied as an augmentation step to the log mel spectrogram of the samples. The algorithm randomly masks 0 to 5 frames in the time domain and 0 to 10 channels in the frequency domain.

5.2 Network Architecture

Face Feature Extractor. ResNet-34 is an important network structure in the field of deep learning and belongs to the ResNet (residual network). This network structure effectively solves the problems of gradient vanishing and representation bottleneck in deep neural networks by introducing residual connections, thus improving the depth of the network and the performance of the model. In the experiments, the use of ResNet-34 was able to achieve good face recognition on the target dataset, proving that the features extracted by this network differed between different people and could be used in this task.

Voice Feature Extractor. ECAPA-TDNN is a neural network model proposed by Desplanques et al. [17], which is mainly used for speaker recognition. ECAPA-TDNN utilizes an extended TDNN-x-vector for voice feature extraction, which consists of multiple frame-level TDNN layers, a statistical pooling layer and two sentence-level fully connected layers, as well as a layer of softmax with a loss function of cross-entropy. ECAPA-TDNN can accept input of any length and merge frame-level features into whole-sentence features. In addition, Wav2vec and HuBERT are examined as voice encoders to assess the robustness of the proposed method. The fine-tuning process is described in detail in Sect. 6.

5.3 Training Process

All models are trained with a step learning rate varying between $1e-8$ and $1e-3$ in conjunction with the Adam optimizer. To prevent over fitting, we apply a weight decay on all weights in the model of $2e-5$. The mini-batch size for training is 64. To prevent false transports, α in L is 0 for the first 2000 epochs, then α is set to 0.5. To find the effect of whether the model is pre-trained, we use pre-trained parameters for the face feature extractor in the training phase. In the results table, learning with pre-trained parameters is marked with ‘*’.

6 Experiment Result and Discussion

6.1 Evaluation Protocols

In the evaluation process, the test data were divided into different groups. Following the criteria in the study [11], the testing process was conducted on seen-heard samples and unseen-unheard samples, under which the test data were further divided into different groups. In order to investigate the effect of specific attributes on this task, these attributes are restricted in the selection of negative test samples, where positive and negative samples in the corresponding grouping need to be consistent in the following attributes: gender (G), nationality (N) and age (A). The gender and nationality labels are taken from Wikipedia, and the age grouping is obtained by applying an age classifier to the face frames in the videos and averaging them for each video.

The correspondance of features from different modalities are evaluated in terms of two tasks: validation and matching, and the specific settings are described below.

Matching. We are were given a probe from one modality along with two alternative samples from another modality one of which was a sample from the same person as the probe, and the other an interfering sample from another person. Our goal is to determine which alternative sample matches the sample. To measure the accuracy of the match, we report the accuracy of the match from voice to face (V-F) and the accuracy of the match from face to voice (F-V), respectively.

Verification. We are given a speech clip and an image of a face. Our task is to determine whether they belong to the same identity. The judgment is based on comparing the similarity of the two input features with a given threshold. Using different thresholds, we can get different false acceptance and false rejection rates. The final performance is reported by the Area Under the ROC curve (AUC).

6.2 Quantitative Results and Discussion

The verification result is shown in Table 2, and the matching result is in Table 3. The experimental results are analysed and discussed below.

Effect of Overall Distributional Alignment. The results of only considering L_c are labelled ‘Contrastive’ and the results of training with overall loss are labelled ‘Overall Align’. Compared to contrastive learning only, the accuracy of face-voice matching and verification of seen individuals has improved. In the verification results, the accuracy of ‘Overall Align’ is better than ‘Contrastive’ in every subgroup. In the groupings with no restrictions (labelled ‘-’), the verification accuracy increased from 60.7 to 69.6 after the addition of the overall distribution, and the matching accuracy of $F \rightarrow V$ increased from 69.4 to 74.7. The comparison of the results is displayed intuitively in Fig. 4. The addition of the overall domain alignment method improves recognition in both tasks compared to the absence

Table 2. Experiment results (Verification). Letters in the table indicate different constraints (e.g. A, G) and ‘*’ indicates that pre-trained parameters were used in the training.

Method	unseen-unheard					seen-heard				
	–	G	N	A	GNA	–	G	N	A	GNA
PINs [11]	78.5	61.1	77.2	74.9	58.8	87.0	74.2	85.9	86.6	74.0
DIMNet [12]	83.2	71.2	81.9	78.0	62.8	94.7	89.8	93.2	94.8	87.8
SSNet [19]	78.8	62.4	53.1	73.5	51.4	91.2	82.5	89.9	90.7	81.8
Contrastive	60.7	55.2	60.7	59.4	54.7	91.5	85.4	90.9	91.4	84.5
Contrastive*	61.2	55.8	61.7	60.2	54.7	92.6	88.0	92.0	92.6	87.0
Overall Align	69.6	61.4	69.1	67.9	59.4	91.7	88.7	91.5	91.7	86.9
Overall Align*	60.9	55.3	61.1	60.2	55.1	94.3	91.0	94.0	94.6	90.4

Table 3. Experiment results (Matching). Letters in the table indicate different constraints (e.g. A, G) and ‘*’ indicates that pre-trained parameters were used in the training.

Method	V → F				F → V			
	–	G	N	GN	–	G	N	GN
LAFV [31]	78.2	62.9	76.4	61.6	78.6	61.6	76.7	61.2
CID [5]	78.3	64.7	77.8	64.2	77.6	64.7	77.2	64.1
Contrastive	71.6	63.2	72.5	61.4	69.4	60.5	70.0	60.9
Contrastive*	72.5	62.1	72.6	62.3	72.4	63.3	72.6	61.4
Overall Align	73.3	63.3	73.2	62.2	74.7	65.5	75.2	64.0
Overall Align*	71.6	62.5	71.9	61.4	72.9	64.2	73.8	62.9

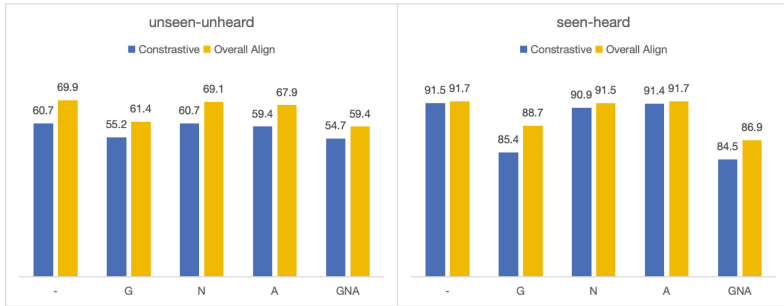
Table 4. Comparison of different vocal encoders. The verification results on unseen-unheard samples is compared.

Model	Method	–	G	N	A	GNA
ECAPA-TDNN	Contrastive	60.7	55.2	60.7	59.4	54.7
	Overall Align	69.6	61.4	69.1	67.9	59.4
wav2vec	Contrastive	81.4	67.8	81.4	78.7	63.7
	Overall Align	84.4	60.4	85.0	81.4	58.4
HuBERT	Contrastive	82.3	71.9	81.6	79.3	66.2
	Overall Align	84.2	64.8	84.6	80.9	60.5

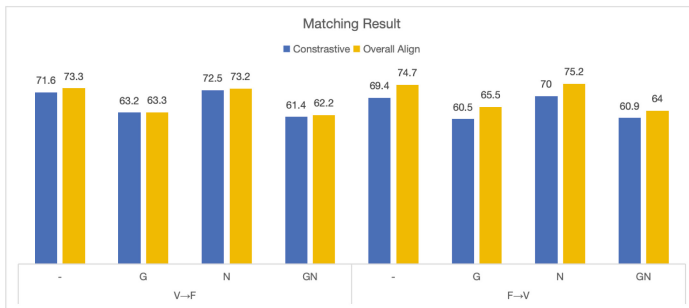
of this method under several different conditions, proving the necessity of overall alignment in the cross-modal feature matching task.

In order to evaluate the robustness of our method, we explored in the experiments more vocal encoders: wav2vec [32] and HuBERT [33], which are two models with state-of-the-art performance in speaker recognition task. The fine tuning process for wav2vec and HuBERT follows the step in the previous study. [34]. The veri-

fication performance of unseen-unheard samples is compared in Table 4. Thanks to the strong ability of the transformer structure, the accuracy has largely increased in each verification group. With Optimal Transport, the verification result in “-”, “N” and “A” groups are all increased, indicating its necessity in this task. The application of wav2vec and HuBERT is quite simple in the experiments, the way to use these encoders needs to be tuned carefully in future works.



(a) Comparison on the verification task



(b) Comparison on the matching task.

Fig. 4. Comparison of ‘Contrastive’ and ‘Overall Align’ on verification and matching task. The overall distribution alignment can effectively improve the results on both tasks under different settings.

Effect of Pre-trained Parameters. The results with and without pre-trained model are compared. The results of learning with pre-trained parameters is labeled with ‘*’ in Table 2 and Table 3. After applying the pre-trained parameters, the verification accuracy of the seen-heard in each experimental group showed an improvement of 1% to 2%. Moreover, in these subgroups, using overall alignment is more effective than using contrastive method alone, whether or not pre-trained parameters are used. The results is displayed intuitively in Fig. 5. The results indicate that pre-trained parameters can improve the search of generic feature space with limited data availability. Further research is needed to determine the details of their use. This component may be utilized to reference knowledge pertinent to zero-shot learning, thereby facilitating further optimization.

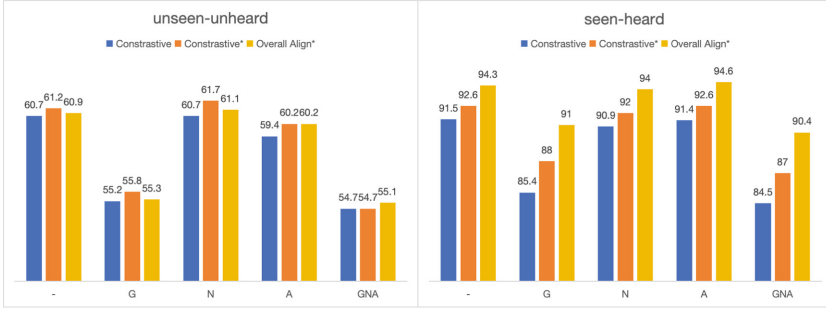


Fig. 5. Comparison of training with and without pre-trained model on verification task. Using pre-trained parameters improves verification results when only the comparison method is used. The overall distribution alignment is useful when pre-trained parameters are used.

Comparison with Prior Study. Compared to the existing methods in the table, our method has an advantage in the final test results. The accuracy of verification and matching on the test group of seen-heard samples with more constraints (‘GNA’ group in the verification task and ‘GN’ group in the matching task) is better than other methods listed in the table. This demonstrates the effectiveness of our method in achieving feature correspondence between two modalities. Furthermore, our method does not require any additional identity information, making it less demanding than existing methods [12].

7 Conclusion

For the unsupervised cross-modal feature matching problem, we add alignment to the overall distribution of features in addition to the direct alignment of corresponding face images and voice features using contrast learning. The experiment results indicate that optimal transport can assist the process of contrast learning by aligning distributions, and has been shown to support the search for a generic space across different modalities. Pre-trained parameters can also be useful in specific situations, but may lead to overfitting issues. Further research should focus on improving the generalisation of the model to improve the recognition of unknown samples. In the future, we will try to improve the results by exploring different network structures, model training methods and model generalisation techniques.

Acknowledgement. Sincere gratitudes are expressed to Mr. Tian Zhang and Mr. Xiaobo Hu for their helps in preparing this paper. This paper was partially financially supported by the Shanghai Cross-Disciplinary Research Fund under grant HXCBCY-2022-054.

References

1. Wells, T., Baguley, T., Sergeant, M., Dunn, A.: Perceptions of human attractiveness comprising face and voice cues. *Arch. Sex. Behav.* **42**, 805–811 (2013)
2. Awwad Shiekh Hasan, B., Valdes-Sosa, M., Gross, J., Belin, P.: “Hearing faces and seeing voices”: a modal coding of person identity in the human brain. *Sci. Rep.* **6**(1), 37494 (2016)
3. Joassin, F., Pesenti, M., Maurage, P., Verreckett, E., Bruyer, R., Campanella, S.: Cross-modal interactions between human faces and voices involved in person recognition. *Cortex* **47**(3), 367–376 (2011)
4. Kamachi, M., Hill, H., Lander, K., Vatikiotis-Bateson, E.: Putting the face to the voice’: matching identity across modality. *Curr. Biol.* **13**(19), 1709–1714 (2003)
5. Morgado, P., Misra, I., Vasconcelos, N.: Robust audio-visual instance discrimination. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12934–12945 (2021)
6. Arandjelovic, R., Zisserman, A.: Look, listen and learn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 609–617 (2017)
7. Xie, Z., Li, L., Zhong, X., Zhong, L.: Image-to-video person re-identification by reusing cross-modal embeddings. arXiv preprint [arXiv:1810.03989](https://arxiv.org/abs/1810.03989) (2018)
8. Oh, T.-H., et al.: Speech2Face: learning the face behind a voice. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7539–7548 (2019)
9. Mercea, O.-B., Riesch, L., Koepke, A., Akata, Z.: Audio-visual generalised zero-shot learning with cross-modal attention and language. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10553–10563 (2022)
10. Zhu, B., et al.: Unsupervised voice-face representation learning by cross-modal prototype contrast. arXiv preprint [arXiv:2204.14057](https://arxiv.org/abs/2204.14057) (2022)
11. Nagrani, A., Albanie, S., Zisserman, A.: Learnable pins: cross-modal embeddings for person identity. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 71–88 (2018)
12. Wen, Y., Ismail, M.A., Liu, W., Raj, B., Singh, R.: Disjoint mapping network for cross-modal matching of voices and faces. arXiv preprint [arXiv:1807.04836](https://arxiv.org/abs/1807.04836) (2018)
13. Nagrani, A., Albanie, S., Zisserman, A.: Seeing voices and hearing faces: cross-modal biometric matching. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8427–8436 (2018)
14. Cheng, K., Liu, X., Cheung, Y.-M., Wang, R., Xu, X., Zhong, B.: Hearing like seeing: improving voice-face interactions and associations via adversarial deep semantic matching network. In: Proceedings of the 28th ACM International Conference on Multimedia, pp. 448–455 (2020)
15. Wang, B., Yang, Y., Xu, X., Hanjalic, A., Shen, H.T.: Adversarial cross-modal retrieval. In: Proceedings of the 25th ACM International Conference on Multimedia, pp. 154–162 (2017)
16. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2019)
17. Desplanques, B., Thienpondt, J., Demuynck, K.: ECAPA-TDNN: emphasized channel attention, propagation and aggregation in TDNN based speaker verification. arXiv preprint [arXiv:2005.07143](https://arxiv.org/abs/2005.07143) (2020)

18. Xiao, Y., Zhou, A., Zhou, L., Zhao, Y.: Automatic insect identification system based on se-resnext. *Int. J. Syst. Control Commun.* **14**(1), 81–98 (2023)
19. Nawaz, S., Janjua, M.K., Gallo, I., Mahmood, A., Calefati, A.: Deep latent space learning for cross-modal mapping of audio and visual signals. In: 2019 Digital Image Computing: Techniques and Applications (DICTA), pp. 1–7. IEEE (2019)
20. Flamary, R., Courty, N., Tuia, D., Rakotomamonjy, A.: Optimal transport for domain adaptation. *IEEE Trans. Pattern Anal. Mach. Intell.* **1**(1–40), 2 (2016)
21. Courty, N., Flamary, R., Habrard, A., Rakotomamonjy, A.: Joint distribution optimal transportation for domain adaptation. In: *Advances in Neural Information Processing Systems*, vol. 30 (2017)
22. Damodaran, B.B., Kellenberger, B., Flamary, R., Tuia, D., Courty, N.: Deepjdot: deep joint distribution optimal transport for unsupervised domain adaptation. In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 447–463 (2018)
23. Zhang, R., et al.: Optimal transport with a diversified memory bank for cross-domain speaker verification. In: *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE (2023)
24. Ge, C., et al.: Domain adaptation via prompt learning. *IEEE Trans. Neural Netw. Learn. Syst.* 1–11 (2023)
25. Aghajanyan, A., Gupta, S., Zettlemoyer, L.: Intrinsic dimensionality explains the effectiveness of language model fine-tuning. 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. 7319–7328, Association for Computational Linguistics (2021)
26. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: *Proceedings of International Conference on Computer Vision (ICCV)* (2015)
27. Cao, Q., Shen, L., Xie, W., Parkhi, O.M., Zisserman, A.: VGGFace2: a dataset for recognising faces across pose and age. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), pp. 67–74. IEEE (2018)
28. Nagrani, A., Chung, J.S., Zisserman, A.: Voxceleb: a large-scale speaker identification dataset. *arXiv preprint [arXiv:1706.08612](https://arxiv.org/abs/1706.08612)* (2017)
29. Peyré, G., Cuturi, M., et al.: Computational optimal transport. In: *Center for Research in Economics and Statistics Working Papers*, no. 2017-86 (2017)
30. Park, D.S., et al.: SpecAugment: a simple data augmentation method for automatic speech recognition. *arXiv preprint [arXiv:1904.08779](https://arxiv.org/abs/1904.08779)* (2019)
31. Kim, C., Shin, H.V., Oh, T.-H., Kaspar, A., Elgharib, M., Matusik, W.: On learning associations of faces and voices. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) *ACCV 2018, Part V. LNCS*, vol. 11365, pp. 276–292. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20873-8_18
32. Baevski, A., Zhou, Y., Mohamed, A., Auli, M.: wav2vec 2.0: a framework for self-supervised learning of speech representations. *Adv. Neural. Inf. Process. Syst.* **33**, 12449–12460 (2020)
33. Hsu, W.-N., Bolte, B., Tsai, Y.-H.H., Lakhotia, K., Salakhutdinov, R., Mohamed, A.: HuBERT: self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM transactions on audio, speech, and language processing* **29**, 3451–3460 (2021)
34. Vaessen, N., Van Leeuwen, D.A.: Fine-tuning wav2vec2 for speaker recognition. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7967–7971. IEEE (2022)



KunquDB: An Attempt for Speaker Verification in the Chinese Opera Scenario

Huali Zhou^{1,2}, Yuke Lin^{1,2}, Dong Liu², and Ming Li^{1,2}(✉)

¹ School of Computer Science, Wuhan University, Wuhan, China
hualizhou@whu.edu.cn

² Suzhou Municipal Key Laboratory of Multimodal Intelligent Systems, Data Science Research Center, Duke Kunshan University, Kunshan, China
{yuke.lin,dong.liu,ming.li369}@dukekunshan.edu.cn, ming.li369@duke.edu

Abstract. This work aims to promote Chinese opera research in both musical and speech domains, with a primary focus on overcoming the data limitations. We introduce KunquDB, <https://hualizhou167.github.io/KunquDB>, a relatively large-scale, well-annotated audio-visual dataset comprising 339 speakers and 128 h of content. Originating from the Kunqu Opera Art Canon (*Kunqu yishu dadian*), KunquDB is meticulously structured by dialogue lines, providing explicit annotations including character names, speaker names, gender information, vocal manner classifications, and accompanied by preliminary text transcriptions. KunquDB provides a versatile foundation for role-centric acoustic studies and advancements in speech-related research, including Automatic Speaker Verification (ASV). Beyond enriching opera research, this dataset bridges the gap between artistic expression and technological innovation. Pioneering the exploration of ASV in Chinese opera, we construct four test trials considering two distinct vocal manners in opera voices: stage speech (*ST*) and singing (*S*). Implementing domain adaptation methods effectively mitigates domain mismatches induced by these vocal manner variations while there is still room for further improvement as a benchmark.

Keywords: Kunqu Opera · Dataset · Multi-modal · Speaker verification · Cross-domain

1 Introduction

Chinese opera, or *Xiqu*, is a distinguishable and traditional art form that has gained worldwide recognition. Kunqu Opera, Beijing Opera, and Cantonese Opera have been proclaimed World Intangible Cultural Heritage, highlighting their exceptional artistic contributions and rich cultural heritage. Chinese opera is a confluence of song, speech, mime, dance, and acrobatics, bound together by theatrical conventions that differ significantly from Western opera [20].

As a distinctive form of performing arts, Chinese opera diverges from conventional speech and typical singing. In the realm of speech research, opera provides a distinctive experimental ground, given its intricate fusion of speech,

music, and theatrical elements. The multifaceted acoustic expressions within opera voices create an exceptional context for in-depth exploration in speech research. Regardless, previous research on Chinese opera has predominantly stemmed from musical and literary perspectives, relying on traditional methodologies rather than integrating state-of-the-art technical tools. The absence of automated deep-learning tools has led to a heavy reliance on manual data pipelines for collecting and annotating Chinese opera datasets. Consequently, existing opera datasets [5, 6, 12, 16] face limitations in terms of scale and annotation richness, typically covering only a few hours [6, 16] and providing genre information exclusively [6]. In contrast to the comprehensive annotations provided in speech and singing datasets, which include speaker labels, text transcriptions, phoneme-level durations, and pitch information, existing Chinese opera datasets lack comparable richness.

The scarcity of detailed annotations poses a significant obstacle for numerous research tasks on opera data. This obstacle is particularly pronounced for tasks requiring comprehensive annotations, including automatic speaker recognition for speaker label prediction, Automatic Speech Recognition (ASR) for text transcription retrieval, speaker diarization for role detection, as well as speech and singing voice synthesis. Meanwhile, speech-related research predominantly focuses on conventional speech. Existing open-source models designed for various speech tasks, such as speaker diarization, exhibit inadequate robustness when applied to opera data. The complex acoustic characteristics in opera voices provide a diverse testing ground for evaluating the robustness of speech models. The absence of automated tools further obstructs large-scale data collection and cleaning, restricting access to diverse and abundant datasets. This dilemma creates a cycle that impedes progress in data availability, hindering the development of advanced tools for digitizing opera research.

In response to the challenges posed by insufficient data and limitations of existing models in the field of Chinese opera, our primary objective is to create a symbiotic relationship between data and models. To achieve this, we present a comprehensive and publicly accessible audio-visual dataset characterized by its richness and scale. This resource is designed to lay the groundwork for developing specialized automated tools applicable to Chinese opera, thereby facilitating advancements in the study of this art form. Narrowing down from the landscape of Chinese traditional opera, we focus on one exquisite domain, Kunqu Opera. Reputed as the mother of Chinese operas, Kunqu Opera boasts a history spanning over 600 years [10], giving rise to numerous operas, including Beijing Opera. In alignment with [5], we selectively choose classic and authoritative audio-visual materials sourced from the Kunqu Opera Art Canon (*Kunqu yishu dadian*)¹ [39] to ensure both quantity and quality. The source video undergoes sentence-level

¹ **Note:** After purchasing the book, we negotiated with the publisher and secured their authorization for its utilization in Kunqu Opera research. The publisher explicitly stated that the book’s digital resource can be employed solely for scholarly or research endeavors upon the approval of the publisher. It may not be illegally disseminated or used for commercial purposes.

segmentation, generating preliminary text transcriptions. Subsequently, we proceed with speaker annotation and explicitly categorize each utterance as either stage speech (*ST*) or singing (*S*) based on vocal manner. Ultimately, KunquDB², the curated audio-visual dataset comprises 339 performers, totals approximately 128 h, with stage speech and singing voices each constituting about half of the dataset. As an audio-visual dataset, it is applicable in various scenarios, including ASV, ASR, speaker diarization, singing voice synthesis, person re-identification and multi-modal understanding.

Building on KunquDB, we investigate automatic speaker verification within the Kunqu Opera context. We aim to provide insights that enhance subsequent synthesis efforts, accommodating variations in role types and vocal manners. Speaker verification in Kunqu Opera bears similarities to the task in [2], involving speech from interviews (typically calm and quiet) and speech from movies (with varying emotion and background noise) across different domains. This yields a cross-domain speaker verification challenge induced by vocal manner, an internal factor of the speaker [23]. To tackle cross-domain issues, we implement domain adversarial training, leveraging domain prediction to obtain speaker-discriminative and domain-invariant representations. Furthermore, we employ the batchwise Siamese training strategy to maintain consistency across different vocal manners for the same speaker. Experimental results validate the efficacy of the domain adaptation methods.

Our main contributions are summarized as follows:

- We curate KunquDB (See Footnote 2), a comprehensive audio-visual dataset specifically tailored for Kunqu Opera. Its large scale effectively mitigates data shortages and fosters a positive feedback loop between data and tool models.
- To the best of our knowledge, we are the first to explore ASV within Chinese opera, addressing mismatches across stage speech and singing. The implementation of domain adaptation methods sets a benchmark for future research.

2 Vocal Distinctions in Chinese Opera Versus Speech

The aural aspect of Chinese traditional opera significantly differs from ordinary spoken and contemporary singing, including textual structure, pronunciation, intonation, vocal manner, and overall expressive forms. (1) The textual dimension of Chinese traditional opera involves two types: song lyrics (*changci*) for expressing emotions and stage speech (*nianbai*) for advancing the narrative [40]. Within the text, two linguistic levels emerge: classical Chinese (*wenyan wen*), an archaic written language, and vernacular (*baihua*), which includes standard spoken Mandarin or regional dialects with distinct phonetic variations. (2) From a melodic perspective, Chinese opera draw its musical compositions from a pre-existing repertoire of tunes. Unlike Western opera, where a designated “composer” is assigned, in Chinese opera, the scriptwriter selects tunes deemed suitable for the dramatic context from the repertoire and crafts the accompanying

² <https://hualizhou167.github.io/KunquDB>.

text. Musical notation is absent; instead, the script specifies tunes by name, with the text intended to be sung accordingly [47]. Notably, stage speech and singing exhibit considerably higher Equivalent Sound Levels (Leq) compared to regular speech [10]. (3) From a vocalization standpoint, Chinese opera utilizes two vocal techniques: “false-voice” (*jiasangzi*), executed in falsetto, and “true-voice” (*zhensangzi*), produced by vocal cord vibration. Falsetto serves various purposes. Firstly, male actors, exemplified by renowned figures like Mei Lanfang, portray female characters in Chinese opera, employing falsetto to imitate the female voice [17]. Secondly, falsetto is believed to ideally produce essential, extended sounds pronounced with a nearly closed mouth [40].

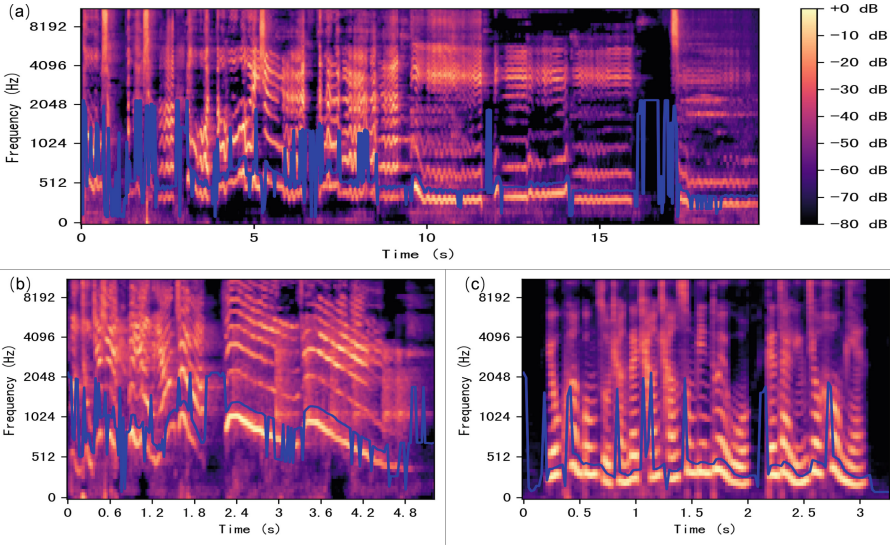


Fig. 1. Mel spectrograms with overlaid pitch contours for singing (a), stage speech (b), and regular speech (c).

Figure 1 displays Mel spectrograms with overlaid pitch contours for randomly selected utterances representing singing, stage speech, and regular speech (from an external speech dataset). Singing and stage speech consistently exhibit higher frequency compared to regular speech. Moreover, singing showcases more dynamic pitch variation than stage speech, highlighting two distinct acoustic characteristics in Chinese opera.

3 Related Works

3.1 Chinese Opera

Open-source datasets for Chinese opera remain limited. While [16] and [6] propose datasets for opera genre and Cantonese singing genre classification, respectively, these datasets are not publicly accessible. Due to the lack of publicly

available corpora, [19, 45, 48] targeting opera genre classification, collect data individually for personalized experiments. Typically, these datasets consist of individual instances structured as audio files paired with corresponding Chinese opera genre labels. On the other hand, open-access datasets have driven advancements in academic research. For example, the CompMusic Beijing Opera corpus proposed in [32] aids [35] in acquiring Beijing Opera percussion patterns for transcription and recognition. Similarly, the unaccompanied singing data released by [1] provides the foundation for [44] to analyze pitch histograms and vibrato statistics in Beijing Opera singing.

Due to the lack of automatic tools for data collection and cleaning, existing opera datasets [5, 6, 12, 16] are limited in scale and annotation richness. Typically spanning only a few hours [6, 16] and offering plain labels [6], they are insufficient for downstream recognition tasks like ASV and singing speech recognition in the field of Chinese opera. In the intersection of opera and speech-related research, most efforts are focused on synthesis, with reliance on the only publicly accessible, well-annotated yet small-scale dataset, “Jingju a cappella singing” [12]. It serves as the basis for subsequent neural network-based opera synthesis by [25, 41, 42, 49]. While [41] and [42] pioneer neural network-based synthesis using the DurIAN [46] framework, [49] introduces OperaSinger, based on the FastSpeech2 [30] framework, exploring novel data augmentations within this small-scale dataset [12]. In a related vein, [25] attempts to transfer popular singers’ timbre to Chinese opera using the VITS [18] model with the same dataset [12].

3.2 Automatic Speaker Verification

Automatic Speaker Verification (ASV) aims to verify whether a given utterance (test utterance) matches the claimed identity by comparing it with the speaker’s known utterance (enrollment utterance). The rise of DNNs in recent years has triggered the evolution of ASV systems from traditional probabilistic models [7, 31] to deep embedding models [9, 34]. A typical DNN-based ASV architecture consists of key components, including: (i) **neural network backbone** [9, 13, 37] (encoder), (ii) **pooling layer** [24, 33, 38] for temporal aggregation, (iii) **loss function** [8, 36] for training optimization, (iv) **scoring strategy** [26] for assessing similarity between embeddings.

The neural network backbone, as the encoder, extracts frame-level features from the input utterance. This backbone has evolved from architectures like 2D Convolutional Neural Networks (CNNs) [37], Time Delay Neural Networks (TDNNs) [9], and Transformers [13]. Currently, 2D CNNs with ResNet [14] are the most widely adopted. The pooling layer aggregates frame-level features into a fixed-length, utterance-level representation, which is then projected linearly to generate the speaker embedding. Common temporal aggregation techniques include average pooling [38], statistical pooling [33] and attentive pooling [24]. The loss function is the optimized objective during training, such as the Additive Margin Softmax (AM-Softmax) [36] and ArcFace [8]. The scoring strategy, or the back-end model, measures the similarity between enrollment and test utterance embeddings for verification. Typically, cosine similarity or Probabilistic Linear Discriminant Analysis (PLDA) [26] are utilized.

Despite significant progress in ASV, speaker embeddings’ robustness falters with domain shifts, facing challenges from real-world variations [23], resulting in performance degradation. For extrinsic factors, [3] and [27] target noise and far-field conditions for more robust voiceprint representation. Addressing internal factors, [29] and [2] investigate cross-age and diverse emotional scenes, respectively, to further enhance the robustness.

4 KunquDB Dataset (See Footnote 2)

To obtain authentic singing data for Kunqu Opera and ensure an ample dataset, we leverage audio-visual materials from the authoritative Kunqu Opera Art Canon (*Kunqu yishu dadian*) (see footnote 1) [39] as reliable sources. The source videos in this collection [39] contain credits, dialogue lines, and information about vocal manner categories (\mathcal{ST} or \mathcal{S}), all of which are hard-coded directly or indirectly.

4.1 Overall: QAs About KunquDB

What is KunquDB? KunquDB is a Kunqu Opera audio-visual dataset derived from videos featuring manual annotations for opera character names, speaker identity (ID) labels, gender information, singing/stage speech category labels, and preliminary text transcriptions.

Why is Manual Labeling Required? Due to the nature of Kun Opera performances, where the entire stage is often captured rather than close-ups of characters, human faces occupy limited space in the frame. Moreover, performers’ heavy makeup and theatrical costumes further obscure facial features, particularly the waist-length beards (*rankou*) [40] worn by male characters typically completely conceal their mouths. Consequently, conventional pipelines, as used in [21, 23], involving face detection, tracking, verification, and audio-video synchronization for mouth movement and speech, are unsuitable for these opera videos.

How to Get KunquDB? The book [39] purchase grants access to the digital source video data in a supplementary disc. It is the user’s responsibility to get the approval from the publisher to conduct research for non-commercial purposes. We provide annotated data, including segment start and end timestamps, along with associated information, such as character names, speaker names, and preliminary text transcriptions. The open-source annotations and processing scripts can be accessed and downloaded online (see footnote 2).

4.2 Data Collection Pipeline

Step 1: Video Segmentation. We utilize VideoSubFinder³, in conjunction with PaddleOCR⁴ to extract hardcoded subtitles from source videos, yielding

³ <https://sourceforge.net/projects/videosubfinder>.

⁴ <https://github.com/PaddlePaddle/PaddleOCR>.

timestamps for each dialogue line and corresponding text transcriptions. Using `ffmpeg`⁵, we then segment the videos into clips based on the acquired timestamps, resulting in individual video clips for each dialogue line.

Step 2: Manual Labeling. The manual annotation process includes categorizing vocal manner and active speaker annotations. Vocal manner annotation is straightforward, with stage speech and singing categorized based on the font style in the original video subtitles. Active speaker annotation is detailed below and is divided into (i) discriminative speaker tag, (ii) tag-character annotation, and (iii) character-performer mapping based on each play. Eventually, the dataset is structured per dialogue line, encompassing all lines delivered by each performer across different plays.

- i We recruit 20 graduate students to assign active speaker tags, each annotating an average of 8.5 h of source videos. Participants use XnView MP⁶ software to tag active speakers for each line while watching the complete source video. They adhere to a naming format like *spk_01* to ensure consistency and avoid repetition within each play. Overlapping speech segments are instructed to be discarded.
- ii Match the active speaker tags akin to *spk_01* obtained in i with the corresponding characters in each play.
- iii Extract character-performer mapping by digitizing the embedded credits in source videos.

Table 1. Dataset statistics for KunquDB

Types of Utterances	Stage Speech	Singing
# of speakers	288 + 50	288 + 1
# of videos	339 + 5	339 + 2
# of utterances	60066	17902
# of hours	67.46	60.88
Avg # of videos per speaker	3	3
Avg # of utterances per speaker	178	62
Avg length of utterances(s)	4.04	12.24

Table 2. Training and test data split

		#Speakers	#Utterances
Training	Stage Speech	200	55889
	Singing	200	16941
Test	Stage Speech	88 + 50	4177
	Singing	88 + 1	961

⁵ <https://ffmpeg.org>.

⁶ <https://www.xnview.com/en/xnviewmp>.

Step 3: Extract Audio from Video. Initially, we use ffmpeg (See footnote 5) to extract 48 kHz stereo audio from video segments, then Spleeter [15] isolates background music, and finally ffmpeg (See footnote 5) downsamples the audio to mono-channel at 16 kHz.

Step 4: Assessment and Recheck. We extract speaker embeddings for individual utterances, using WeSpeaker’s [37] ResNet34-based model pretrained on Cn-Celeb [11]. Then, we compute average embeddings for each speaker in each category and assess the cosine similarity between each utterance’s embedding and the corresponding average. Utterances with a similarity score below the threshold of 0.4 undergo manual review.

4.3 Dataset Statistics

Table 1 summarizes key statistics for the KunquDB dataset, differentiating stage speech (ST) and singing (S) categories. The dataset contains 60,066 ST utterances and 17,902 S utterances, contributed by 288 speakers for both ST and S data, 50 exclusively providing ST data, and 1 exclusively offering S data. Additionally, there are 339 videos featuring both ST and S , along with 5 exclusively for ST and 2 for S . Figure 2 visually represents the distribution of utterance lengths and speakers enacting role types.

4.4 Split: Training and Test

We divide speakers based on their total number of utterances, with the initial 200 individuals allocated to the training set and the remaining 139 to the test set. See Table 2 for details.

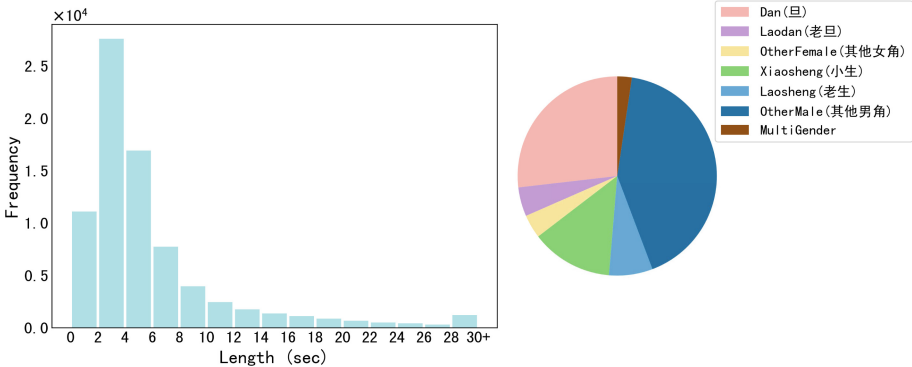


Fig. 2. Left: Histogram of utterance lengths in the dataset. Right: Distribution of speaker role type information. The legend indicates the role type performed by speakers throughout the dataset. *Dan* for young female characters, *LaoDan* for old female characters, *OtherFemale* for additional female characters; *XiaoSheng* for young male characters, *LaoSheng* for old male characters, *OtherMale* for additional male characters; and *MultiGender* means speakers portraying characters of both genders.

4.5 Trial Construction

When generating test trials for speaker verification experiments, we adopt a consistent procedure for each utterance, randomly selecting five positive and five negative samples. Investigating four trial scenarios considering two vocal manners (stage speech and singing), we have:

- Undifferentiated Trial: No distinction between enrollment and test utterance regarding vocal manner categories; samples are randomly chosen from either stage speech or singing.
- Stage Speech Domain Trial: Both enrollment and test utterances are from the stage speech category.
- Singing Domain Trial: Both enrollment and test utterances are from the singing category.
- Cross-domain Trial: Enrollment is from singing, while test utterances are from the stage speech category.

5 Learning Domain-Invariant Speaker Embeddings

5.1 Domain Discrepancy Adversarial Learning

As discussed in Sect. 3.2, the speaker ID embedding extractor comprises a feature encoder, pooling, and linear layer. Traditionally, it is assumed that this extractor, depicted by the pink dashed box in Fig. 3, exclusively captures acoustic features defining speaker identity, denoted by the equation $\mathbf{f} = \mathbf{f}_{id}$. However, it may inadvertently conflate identity-specific traits with variations from intrinsic factors like vocal mannerisms, formalized as Eq. 1, where \mathbf{f} denotes the extracted

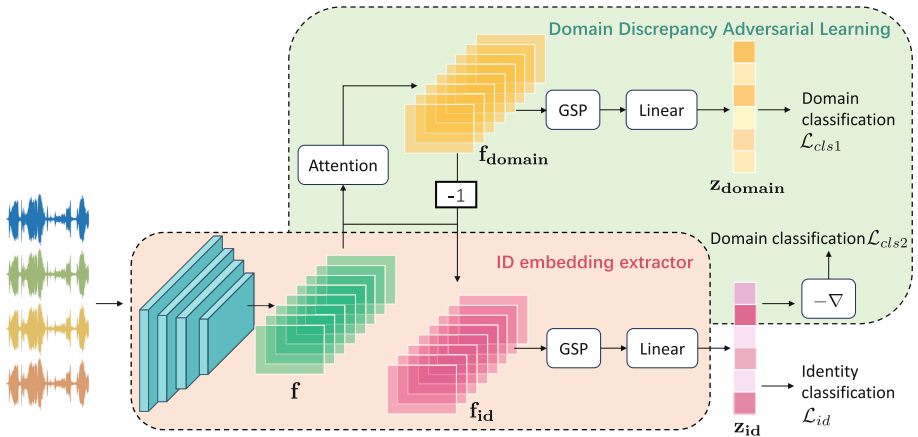


Fig. 3. Schematic of the DDAL framework. The pink dashed box outlines the identity embedding extractor; the green dashed box highlights the core components of the DDAL mechanism. (Color figure online)

features, \mathbf{f}_{id} refers to the identity-specific features, and \mathbf{f}_{domain} represents features associated with vocal manners.

$$\mathbf{f} = \mathbf{f}_{id} + \mathbf{f}_{domain} \quad (1)$$

Borrowing insights from [29], we implement an optimized multi-task paradigm called Domain Discrepancy Adversarial Learning (DDAL), as illustrated in Fig. 3, to isolate domain-specific variables from speaker embeddings. This framework integrates speaker identity verification, domain classification, and domain adversarial training. Diverging from [29], we disentangle domain characteristics at the feature map layer instead of the abstract embedding space. This early disentanglement capitalizes on the richer domain-specific details in the feature map layer, facilitating a cleaner separation and enhancing verification precision across domains.

We leverage an attention mechanism to disentangle domain-related features \mathbf{f}_{domain} induced by different vocal manners from the feature map \mathbf{f} extracted by the backbone model. Next, we refine speaker-specific features, \mathbf{f}_{id} , by filtering out \mathbf{f}_{domain} . Following this, both \mathbf{f}_{domain} and \mathbf{f}_{id} undergo pooling and fully connected layers, producing the domain embedding \mathbf{z}_{domain} for domain classification and speaker ID embedding \mathbf{z}_{id} for speaker classification. Further, we employ a gradient reversal layer (GRL) before an auxiliary domain classifier to eliminate domain influence from \mathbf{z}_{id} through adversarial learning.

Equation 2 defines the composite loss function, comprising the standard identity loss \mathcal{L}_{id} and the weighted sum of domain classifier losses \mathcal{L}_{cls1} and \mathcal{L}_{cls2} . The weight λ_{ddal} acts as a tuning hyperparameter to balance these components:

$$\mathcal{L}_{DDAL} = \mathcal{L}_{id} + \lambda_{ddal}(\mathcal{L}_{cls1} + \mathcal{L}_{cls2}) \quad (2)$$

5.2 Batchwise Contrastive Siamese Training

To effectively utilize utterances from the same speakers, we adopt a Batchwise Contrastive Siamese Training (BCST) strategy, inspired by [22], to refine speaker embeddings across different domains into a unified, domain-independent representation. As depicted in Fig. 4, the model receives paired utterances from the same speaker but in different vocal manners.

The optimization process focuses on the \mathcal{L}_{BCST} , a combined loss comprising the individual utterance losses, \mathcal{L}_{uttS} and \mathcal{L}_{uttST} , as well as the pair loss \mathcal{L}_{pair} scaled by a factor λ_{bcst} . The pair loss quantifies the cosine distance between speaker embeddings, \mathbf{z}_{id}^{ST} and \mathbf{z}_{id}^S , extracted from paired utterances. By leveraging both singular utterance traits and relational information from utterance pairs, the model is encouraged to enhance its ability to distinguish between speakers and maintain feature consistency for the same speaker, even when their vocal manner varies.

$$\mathcal{L}_{BCST} = \mathcal{L}_{uttS} + \mathcal{L}_{uttST} + \lambda_{bcst}\mathcal{L}_{pair} \quad (3)$$

$$\mathcal{L}_{pair} = 1 - \frac{\mathbf{z}_{id}^{ST} \cdot \mathbf{z}_{id}^S}{\|\mathbf{z}_{id}^{ST}\| \|\mathbf{z}_{id}^S\|} \quad (4)$$

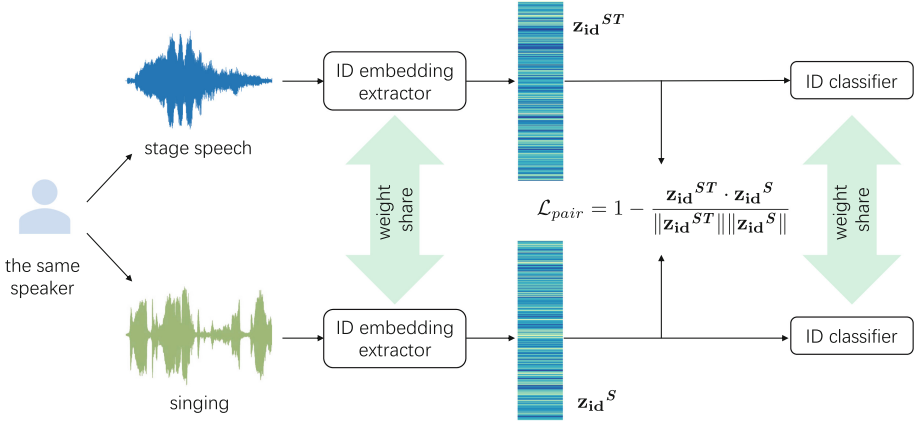


Fig. 4. Overview of the BCST structure

6 Experiments

6.1 Experimental Setup

Dataset. We pretrain the model on VoxBlink2 [21] with over 16,000 h of audio data from 110k speakers. Thereupon, we fine-tune the model using KunquDB’s training set. Evaluation is performed on the KunquDB test set.

Network. In our baseline (detailed in Table 3), we use ResNet34 [14] as the feature extractor, followed by a Global Statistic Pooling (GSP) layer to condense the length-variable frame-level feature map into a fixed-length representation. This representation is then input to a fully connected layer with 256 dimensions. For speaker identification, we employ the ArcFace classifier [8] ($m = 0.2$, $s = 32$). Binary domain classifier involves stacking Linear-ReLU-Linear structures on $\mathbf{z}_{\text{domain}}$ and \mathbf{z}_{id} for domain classification and adversarial learning, respectively. In the attention mechanisms that decouple domain-related features $\mathbf{f}_{\text{domain}}$ from global features \mathbf{f} , we employ two approaches: a neural network-based method known as Attentive Statistics Pooling (ASP) [24] and a Simple, Parameter-free Attention Module (SimAM) [43].

We initialize the baseline model by pre-training on the VoxBlink2 dataset and experiment with various fine-tuning strategies using the KunquDB training set, as detailed in Table 4. **M0** serves as the standard and starting point for all subsequent fine-tuning experiments; it is pre-trained but not fine-tuned. **M1** and **M2** undergo fine-tuning using the standard ResNet34-GSP architecture, aligning with **M0**. In contrast, **M3** and **M4** are built on the SimAM-based DDAL framework; likewise, **M5** and **M6** adopt the ASP-based DDAL approach. **M2**, **M4**, and **M6** incorporate the BCST strategy, further building upon **M1**, **M3**, and **M5**, respectively.

Table 3. The architecture of our ResNet34 backbone network. The residual building blocks are shown in $[\cdot]$, with the numbers of blocks stacked. Downsampling is performed by Layer2_1, Layer3_1, Layer4_1 with a stride of 2.

Layer	Structure	Output Size
Conv1	$3 \times 3, 64$	$64 \times 80 \times T$
Layer1	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$64 \times 80 \times T$
Layer2	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$128 \times 40 \times \frac{T}{2}$
Layer3	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$256 \times 20 \times \frac{T}{4}$
Layer4	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$512 \times 10 \times \frac{T}{8}$
Encoding	Global Statistics Pooling	1024
ID Embedding	Linear	256
Domain Embedding	Linear	256

Table 4. Models varied in architectures, training data, and strategies. **KunquDB fine-tuning** indicates whether to utilize the KunquDB training set for fine-tuning. **DDAL** denotes Domain Discrepancy Adversarial Learning as described in Sect. 5.1; **BCST** refers to Batchwise Contrastive Siamese Training as detailed in Sect. 5.2.

ID	Model	Size	KunquDB fine-tuning	BCST
M0	ResNet34-GSP	20.54M	\times	\times
M1			\checkmark	\times
M2			\checkmark	\checkmark
M3	+ SimAM-based DDAL	20.79M	\checkmark	\times
M4			\checkmark	\checkmark
M5	+ ASP-based DDAL	27.35M	\checkmark	\times
M6			\checkmark	\checkmark

Training Details. During pre-training, we apply on-the-fly data augmentation [4] and follow a training setting similar to [28]. For fine-tuning, we utilize a multi-step learning rate (LR) scheduler starting with an initial LR of 10^{-3} to modulate the SGD optimizer, gradually updating the model parameters until convergence. The hyperparameters λ_{ddal} and λ_{bcst} are assigned with a value of 0.5 when used independently within the model (**M1**, **M2**, **M3**, **M5**). However, when both are employed (**M4**, **M6**), λ_{ddal} is set to 1, while λ_{bcst} is adjusted to 1.5. Input utterances are truncated to 2s and converted to 80-dimensional log Mel-filterbank energies.

Table 5. The performance comparison of different speaker verification systems in terms of Equal Error Rate (EER) across four distinct test sets, as outlined in Sect. 4.5.

ID	Undifferentiated		<i>ST</i> -Domain		<i>S</i> -Domain		Cross-Domain	
	EER [%]	mDCF	EER [%]	mDCF	EER [%]	mDCF	EER [%]	mDCF
M0	21.48	0.99	18.81	0.97	23.06	0.97	28.52	1.00
M1	7.95	0.66	7.53	0.61	7.29	0.77	9.84	0.84
M2	7.79	0.67	7.67	0.65	6.47	0.70	9.37	0.79
M3	7.79	0.71	7.57	0.64	7.20	0.87	9.40	0.88
M4	7.36	0.71	7.12	0.70	6.21	0.72	8.37	0.84
M5	7.64	0.71	7.56	0.63	6.41	0.78	8.79	0.88
M6	7.39	0.69	7.41	0.63	6.32	0.71	8.25	0.78

Evaluation Metrics. Cosine similarity is used for trial scoring. The verification performances are measured by the Equal Error Rate (EER) and the minimum normalized detection cost function (mDCF) with $P_{target} = 0.01$.

Experimental Results. Table 5 reports the performance of models on different test sets, with several key observations discussed below.

- (1) Model **M0** shows weak robustness on Kunqu data, performing best in the *ST*-domain due to its exclusive pretraining on speech data. Nevertheless, its performance is still markedly inferior to its excellent performance on regular speech test sets, often below 1% EER.
- (2) Models generally perform best when enrollment and test utterances share the same vocal manner, whether in the *S* or *ST* category. However, their performance notably declines in cross-domain scenarios, indicating the difficulty in extracting domain-agnostic speaker embeddings.
- (3) DDAL or BCST individually improves model performance on Kunqu datasets. Deploying both approaches concurrently (**M4** and **M6**) substantially augments this enhancement, delivering superior outcomes.
- (4) Regarding the two implementations of attention within the DDAL strategy, the ASP-based implementation (**M5**) outperforms the SimAM-based counterpart (**M3**) across all test sets without BCST. However, with BCST integration, the SimAM-based approach (**M4**) yields better results than the ASP-based method (**M6**) in three out of four test sets, except for the cross-domain scenario.

We randomly select eleven individuals from the test data and visualize their speaker embeddings using the t-distributed stochastic neighbor embedding (t-SNE) algorithm in Fig. 5. Each subfigure corresponds to a specific model (**M0**–**M6**), providing a visual representation of the distribution patterns learned under various domain adaptation approaches. Notably, the **M0** subfigure reveals a lack of convergence in the distributions of utterances from the same speaker across different domains. In contrast, coherent distributions are observed among similar

utterance types, with *S* utterances predominantly in the left upper quadrant and *ST* utterances in the right lower quadrant. These t-SNE visualizations consistently mirror the objective performance metrics presented in Table 5, confirming the effectiveness of the domain adaptation methods.

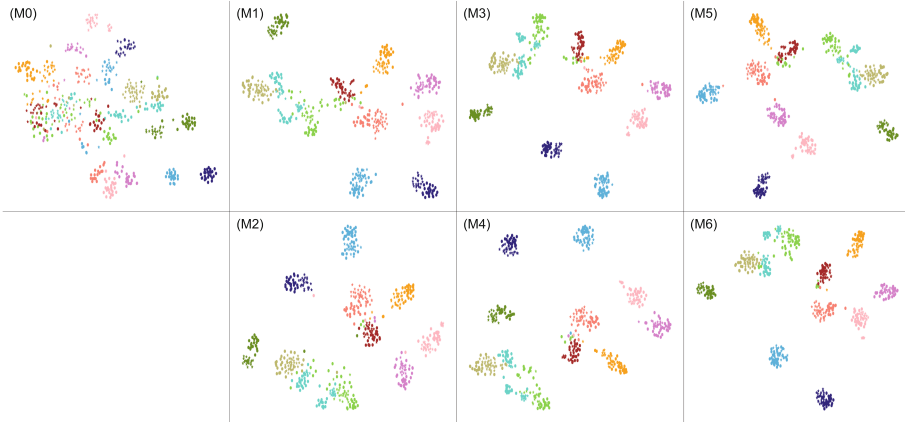


Fig. 5. t-SNE visualization of speaker embedding extracted by seven models (M0–M6). Unique colors signify individual distinctions, with circular markers (●) representing stage speech utterances and pentagonal stars (★) denoting singing utterances.

7 Conclusion

This paper introduces KunquDB, a relatively large-scale, publicly accessible audio-visual dataset designed to address research gaps in Chinese opera studies. With detailed annotations, KunquDB aims to serve as a valuable resource for opera and speech-related research endeavors. Leveraging domain discrepancy adversarial learning and batchwise contrastive Siamese training, we establish benchmarks for ASV on Chinese opera data, offering unique insights distinct from conventional speech datasets.

Acknowledgements. This research is funded by the Kunshan Municipal Government Research Funding under the project “Deep Learning based Singing Voice Synthesis for Kun Opera”. We want to thank the publisher for allowing us to conduct research on their data and DKU library staff members for their coordination. Special thanks to Xiaoyi Qin for his assistance.

References

1. Black, D.A., Li, M., Tian, M.: Automatic identification of emotional cues in Chinese opera singing. In: ICMPC, Seoul, South Korea (2014)

2. Brown, A., Huh, J., Nagrani, A., Chung, J.S., Zisserman, A.: Playing a part: speaker verification at the movies. In: Proceedings of the ICASSP, pp. 6174–6178 (2021)
3. Cai, D., Cai, W., Li, M.: Within-sample variability-invariant loss for robust speaker recognition under noisy environments. In: Proceedings of the ICASSP, pp. 6469–6473 (2020)
4. Cai, W., Chen, J., Zhang, J., Li, M.: On-the-fly data loader and utterance-level aggregation for speaker and language recognition. *IEEE/ACM Trans. Audio Speech Lang. Process.* **28**, 1038–1051 (2020)
5. Caro Repetto, R., Serra, X.: Creating a corpus of jingju (Beijing opera) music and possibilities for melodic analysis. In: Proceedings of the ISMIR (2014)
6. Chen, Q., Zhao, W., Wang, Q., Zhao, Y.: The sustainable development of intangible cultural heritage with AI: cantonese opera singing genre classification based on cogcnet model in china. *Sustainability* **14**(5), 2923 (2022)
7. Dehak, N., Kenny, P.J., Dehak, R., Dumouchel, P., Ouellet, P.: Front-end factor analysis for speaker verification. *IEEE Trans. Audio Speech Lang. Process.* **19**(4), 788–798 (2010)
8. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: additive angular margin loss for deep face recognition. In: Proceedings of the CVPR, pp. 4690–4699 (2019)
9. Desplanques, B., Thienpondt, J., Demuynck, K.: ECAPA-TDNN: emphasized channel attention, propagation and aggregation in tdnn based speaker verification. In: Proceedings of the Interspeech, pp. 3830–3834 (2020)
10. Dong, L., Sundberg, J., Kong, J.: Loudness and pitch of Kunqu opera. *J. Voice* **28**(1), 14–19 (2014)
11. Fan, Y., et al.: CN-CELEB: a challenging Chinese speaker recognition dataset. In: Proceedings of the ICASSP, pp. 7604–7608 (2020)
12. Gong, R., Caro, R., Zhu, T.: Jingju a cappella recordings collection (2019). <https://doi.org/10.5281/zenodo.3251761>
13. Han, B., Chen, Z., Qian, Y.: Local information modeling with self-attention for speaker verification. In: Proceedings of the ICASSP, pp. 6727–6731 (2022)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the CVPR, pp. 770–778 (2016)
15. Hennequin, R., Khlif, A., Voituret, F., Moussallam, M.: Spleeter: a fast and efficient music source separation tool with pre-trained models. *J. Open Sour. Softw.* **5**(50), 2154 (2020)
16. Islam, R., Xu, M., Fan, Y.: Chinese traditional opera database for music genre recognition. In: Proceedings of the O-COCOSDA/CASLRE, pp. 38–41 (2015)
17. Jinpei, H.: Xipi and erhuang of Beijing and Guangdong operas. *Asian Music* **20**(2), 152–195 (1989)
18. Kim, J., Kong, J., Son, J.: Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. In: Proceedings of the ICML, pp. 5530–5540 (2021)
19. Li, Q., Hu, B.: Joint time and frequency transformer for Chinese opera classification. In: Proceedings of the Interspeech (2023)
20. Lin, L.: Modernising Cantonese opera through contemporary sound production design. Ph.D. thesis, Middlesex University (2022)
21. Lin, Y., Cheng, M., Zhang, F., Gao, Y., Zhang, S., Li, M.: VoxBlink2: a 100k+ speaker recognition corpus and the open-set speaker-identification benchmark. arXiv preprint [arXiv:2407.11510](https://arxiv.org/abs/2407.11510) (2024)
22. Lin, Y., Qin, X., Jiang, N., Zhao, G., Li, M.: Haha-pod: an attempt for laughter-based non-verbal speaker verification. In: Proceedings of the ASRU, pp. 1–7 (2023)

23. Nagrani, A., Chung, J., Zisserman, A.: Voxceleb: a large-scale speaker identification dataset. In: *Proceedings of the Interspeech* (2017)
24. Okabe, K., Koshinaka, T., Shinoda, K.: Attentive statistics pooling for deep speaker embedding. arXiv preprint [arXiv:1803.10963](https://arxiv.org/abs/1803.10963) (2018)
25. Peng, Z., Wu, J., Li, Y.: Singing voice conversion between popular music and Chinese opera based on ViTs. In: *Proceedings of the DASC/PiCom/CBDCCom/CyberSciTech*, pp. 0999–1003 (2023)
26. Prince, S.J., Elder, J.H.: Probabilistic linear discriminant analysis for inferences about identity. In: *Proceedings of the ICCV*, pp. 1–8 (2007)
27. Qin, X., Cai, D., Li, M.: Robust multi-channel far-field speaker verification under different in-domain data availability scenarios. *IEEE/ACM Trans. Audio Speech Lang. Process.* **31**, 71–85 (2022)
28. Qin, X., et al.: The DKU-tencent system for the voxceleb speaker recognition challenge 2022. arXiv preprint [arXiv:2210.05092](https://arxiv.org/abs/2210.05092) (2022)
29. Qin, X., Li, N., Weng, C., Su, D., Li, M.: Cross-age speaker verification: learning age-invariant speaker embeddings. In: *Proceedings of the Interspeech* (2022)
30. Ren, Y., et al.: FastSpeech 2: fast and high-quality end-to-end text to speech. In: *Proceedings of the ICLR* (2020)
31. Reynolds, D.A., Quatieri, T.F., Dunn, R.B.: Speaker verification using adapted gaussian mixture models. *Digit. Signal Process.* **10**(1–3), 19–41 (2000)
32. Serra, X.: Creating research corpora for the computational study of music: the case of the compmusic project. In: *Audio Engineering Society Conference: 53rd International Conference: Semantic Audio* (2014)
33. Snyder, D., Garcia-Romero, D., Povey, D., Khudanpur, S.: Deep neural network embeddings for text-independent speaker verification. In: *Proceedings of the Interspeech*, vol. 2017, pp. 999–1003 (2017)
34. Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S.: X-vectors: robust DNN embeddings for speaker recognition. In: *Proceedings of the ICASSP*, pp. 5329–5333 (2018)
35. Srinivasamurthy, A., Caro Repetto, R., Sundar, H., Serra, X.: Transcription and recognition of syllable based percussion patterns: the case of Beijing opera. In: *Proceedings of the ISMIR*, pp. 431–436 (2014)
36. Wang, F., Cheng, J., Liu, W., Liu, H.: Additive margin softmax for face verification. *IEEE Signal Process. Lett.* **25**(7), 926–930 (2018)
37. Wang, H., et al.: WeSpeaker: a research and production oriented speaker embedding learning toolkit. In: *Proceedings of the ICASSP*, pp. 1–5 (2023)
38. Wang, S., Yang, Y., Qian, Y., Yu, K.: Revisiting the statistics pooling layer in deep speaker embedding learning. In: *Proceedings of the ISCSLP*, pp. 1–5 (2021)
39. Wang, W.: *Kunqu yishu dadian (昆曲艺术大典)*. Anhui Literature and Art Publishing House, Anhui (2016). <http://www.awpub.com/front/book/10-858>
40. Wichmann, E.: *Listening to Theatre: The Aural Dimension of Beijing Opera*. University of Hawaii Press (1991)
41. Wu, Y., et al.: Synthesising expressiveness in Peking opera via duration informed attention network. arXiv preprint [arXiv:1912.12010](https://arxiv.org/abs/1912.12010) (2019)
42. Wu, Y., et al.: Peking opera synthesis via duration informed attention network. In: *Proceedings of the Interspeech* (2020)
43. Yang, L., Zhang, R.Y., Li, L., Xie, X.: SimAM: a simple, parameter-free attention module for convolutional neural networks. In: *Proceedings of the ICML*, pp. 11863–11874 (2021)
44. Yang, L., Tian, M., Chew, E., et al.: Vibrato characteristics and frequency histogram envelopes in Beijing opera singing (2015)

45. Yao, M., Liu, J.: The analysis of Chinese and Japanese traditional opera tunes with artificial intelligence technology based on deep learning. *IEEE Access* (2024)
46. Yu, C., et al.: DurIAN: duration informed attention network for multimodal synthesis. arXiv preprint [arXiv:1909.01700](https://arxiv.org/abs/1909.01700) (2019)
47. Yung, B.: Creative process in cantonese opera iii: the role of padding syllables. *Ethnomusicology* **27**(3), 439–456 (1983)
48. Zhang, H., Jiang, Y., Zhao, W., Jiang, T., Hu, P., Entertainment, T.M.: Chinese opera genre investigation by convolutional neural network. In: *Proceedings of the ISMIR* (2021)
49. Zhou, X., Sun, W., Shi, X.: A high-quality melody-aware Peking opera synthesizer using data augmentation. In: *Proceedings of the ICME*, pp. 1092–1097 (2023)



Act-ChatGPT: Introducing Action Features into Multi-modal Large Language Models for Video Understanding

Yuto Nakamizo[✉] and Keiji Yanai[✉]

The University of Electro-Communications, Chohu, Tokyo, Japan
{nakamizo-y, yanai}@mm.inf.uec.ac.jp

Abstract. In the last few years, the advancement of GPT-4 and similar extensive large language models has significantly influenced video comprehension fields, models have been developed to exploit these advances to enhance interactive video comprehension. However, existing models generally encode video using image language models or video language models with sparse sampling, overlooking the vital action features present in each video segment. To address this gap, we propose Act-ChatGPT, an innovative interactive video comprehension model that integrates action features. Act-ChatGPT incorporates a dense sampling-based action recognition model as an additional visual encoder, enabling it to generate responses that consider the action in each video segment. Comparative analysis reveals Act-ChatGPT superiority over a base model, with qualitative evidence highlighting its adeptness at recognizing actions and responding based on them.

Keywords: Multi-Modal Large Language Model · Action Features · Video Understanding · Dual-Encoder strategy

1 Introduction

The evolution of Large Language Models (LLMs) in natural language processing has led to invention of multi-modal LLMs, combining a visual encoder with LLM for enhanced video understanding. This fusion projects visual features onto LLM token spaces, facilitating interactive comprehension. Nevertheless, such models typically use an image language model as a visual encoder or a video language model that is conscious of modeling the entire video, neglecting detailed actions within video segments. Conversely, with the adoption of Transformer [20] and self-supervised learning in video domain, especially models pre-trained on extensive video data, has significantly improved action recognition. These models have high action recognition performance, and in particular, by using models that operate on individual video segments, it is possible to extract good action features from each segment of the video.

Therefore, we propose Act-ChatGPT, an advanced multi-modal LLM tailored for video understanding, which emphasizes the utilization of action features within each video segment. Act-ChatGPT enhances video comprehension by incorporating an action recognition model as an additional visual encoder. This model, designed to extract action features from each video segment, works in tandem with Video-ChatGPT’s existing image-based visual encoder. Moreover, Act-ChatGPT is different from traditional models by adopting a dual-encoder strategy. This approach combines the object recognition strengths of the visual language model with the nuanced human action detection of the action recognition model, enabling a richer video understanding. Our contributions are (1) We propose Act-ChatGPT, which is the first multi-modal LLM for video understanding that introduces action features within each video segment. (2) The experimental results showed the effectiveness of our proposed method by outperforming the baseline, Video-ChatGPT.

2 Related Works

2.1 LLMs

A language model that has been pre-trained by self-supervised learning with a large corpus is called a pre-trained language model. Recently, based on the knowledge that scaling the model parameters and training data of these pre-trained language models can improve the performance of downstream tasks [10], large pre-trained language models with a very large number of parameters and trained on particularly large amounts of data have been constructed. Because these models have an emergent abilities [26] that has not been seen in small-scale pre-trained language models, and because they show tremendous ability in solving a series of complex tasks, they are distinguished from small-scale pre-trained language models and are referred to as LLMs [24]. LLMs excel in their ability to generate language and make common sense inferences, and their use has been studied in many fields, not only in the field of natural language processing but also in other fields. For example, OpenAI’s GPT-4, which has been reported to have particularly excellent instruction response performance, is used for dataset creation, filtering, and data augmentation, because it can be utilized via API. Since LLaMA [8] and its successor, Llama-2 [7], are the LLMs whose models and weights are publicly available, they have become the basis for many LLMs such as Vicuna [3].

Our study delves into utilizing LLMs within the visual domain, particularly focusing on enhancing video understanding through the integration of action features, marking a significant step forward in interactive video understanding.

2.2 Multi-modal LLMs

Current multi-modal LLMs in the visual sphere fall into two primary categories. The first involves leveraging LLMs to interlink specialized models for diverse

visual tasks, exemplified by Visual ChatGPT [2], a system that integrates numerous expert models through a LLM. This setup allows the LLM to process user commands and visual inputs, activating necessary external visual models to fulfill these commands.

The second category involves the methods that merge visual models with LLMs by mapping visual encoder-extracted features onto the LLM’s token space, creating a unified model capable of end-to-end learning. BLIP-2 [12] is included in this category, that employs a “Q-former” module that aimed to bridge the gap between the visual encoder’s features and the LLM’s tokens through end-to-end training using image-text contrast learning, image-text matching and image grounded text generation. Additionally, this category includes LLaVA [5], which introduced Instruction Tuning [22] that is used in the field of natural language processing for visual contexts as Visual Instruction Tuning. This technique enhances instruction-following abilities by fine-tuning LLMs with data composed of instructional texts and their corresponding responses, where visual features are embedded into the instructional content.

In our study, we focus on the latter method and define the latter as Vision-LLM, and the Vision-LLM focusing on the video domain is defined as Video-LLM.

2.3 Video-LLMs

Current Video-LLMs fall into two main categories based on their approach to video encoding: frame-by-frame encoding using an image language model and holistic video encoding using a video language model.

The former-type models, such as VideoChat [13], Video-LLaMA [4], VideoChatGPT [19], and LLaMA-VID [17], encode videos frame by frame. They employ the image language model, CLIP [1], as a visual encoder to extract features from individual frames sampled across the video. These features are often condensed and temporally modeled throughout the entire video using pooling and additional modules before being integrated into the LLM’s token space via a linear layer.

Conversely, the latter-type models, such as VideoChat2 [14] and Video-LLaVA [18], encode videos as a whole. Some video language models such as UMT [16] and LanguageBind [25] capture video-wide features from a limited sampling of 4–16 frames for efficiency. These features are especially focusing on the video’s overall context rather than the detailed temporal elements contained in each segment of the video.

Therefore, the existing Video-LLMs do not explicitly model the temporal features of the video or focus on modeling throughout the entire video and do not focus on the action in each segment of the video. Our study differs from the existing methods in that we introduce action features in each segment of the video to Video-LLM.

2.4 Action Recognition Model

The recent advancements in self-supervised learning have underscored its effectiveness, particularly with transformer-based models such as VideoMAEv2 [21] and UMT [16]. These models, pre-trained on extensive video datasets, have shown remarkable efficacy in action recognition tasks by fine-tuning.

Current action recognition models predominantly fall into two categories based on their frame sampling techniques. The first employs dense sampling, a method that extracts multiple video segments of a set frame length throughout the video, exemplified by VideoMAE v2. The second utilizes sparse sampling, a strategy that selects a fixed number of frames about 4 or 16 from the entire video, regardless of its length, as seen in models like UMT [16]. Dense sampling is suitable for capturing detailed features within individual video segments, while sparse sampling is suitable for providing a broader overview of features across the entire video. Those approach, therefore, offers different unique advantages for modeling action content, in that they either focus on specific segments or the video as a whole.

3 Method

3.1 Overview

We introduce a novel Video-LLM into Video-ChatGPT [19] by integrating action features. Figure 1 provides an overview of our method.

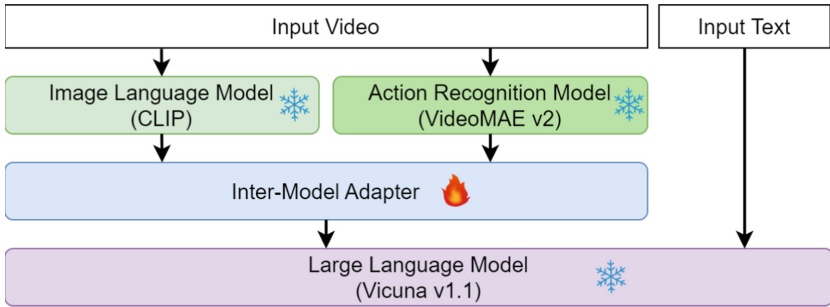


Fig. 1. The overview of Act-ChatGPT

We employ a dual-encoder strategy for the visual encoder, combined using an image language model for frame-based image feature extraction with an action recognition model dedicated to capturing action features from video segments. Initially, we sample T frames, $F \in \mathbb{R}^{T \times W \times H \times C}$, and T sets of 16-frame video segments, $S \in \mathbb{R}^{T \times 16 \times W \times H \times C}$, from the input video. Then, from these samples image features, $V_f \in \mathbb{R}^{T \times N \times D_f}$, and action features, $V_s \in \mathbb{R}^{T \times D_s}$, are extracted via their respective encoders. Here, D_f and D_s represent the dimensional of

the embedded features from the image language model and the action recognition model, respectively. N denotes the number of the image language model’s patches, calculated as $N = W/p \times H/p$ based on the patch size p of the image language model where W, H , and C represent the width, the height, and the channel of the input video.

Subsequently, the extracted image and action features, V_f and V_s , are converted into visual tokens, $Q_v \in \mathbb{R}^{(2T+N) \times D_h}$. Here, D_h represents the dimension of the LLM’s token space. This is achieved through an Inter-Model Adapter that projects each feature set into the LLM’s token space and merges them. The specifics of this conversion process within the Inter-Model Adapter are detailed further in Sect. 3.3.

In the final step, the next tokens are predicted from a visual token, Q_v , and a linguistic token, Q_t , tokenized from the input text, and then a response text is generated by LLM. To optimize training efficiency, in our proposed method, we leverage pre-trained models for both two visual encoders and the LLM and train only the Inter-Model Adapters.

3.2 Using Trained Models

Our method incorporates several pre-trained models across a visual language model, an action recognition model, and LLM components. Initially, for the visual language model, we utilize the OpenAI CLIP [1] ViT-L/14 model. Here, the outputs from the penultimate layer are harnessed as the image features. Secondly, as an action recognition component, we employ the VideoMAEv2 [21] ViT-g/14 model, which has been fine-tuned on the Kinetics-710 dataset [15]. For this model, the action features are derived by applying Layer Normalization to the final layer’s output and calculating the mean value. Lastly, for the LLM, we use Vicuna v1.1 [3], a 7B model fine-tuned for the multi-modal model LLaVA [5].

3.3 Inter-model Adapter

Figure 2 provides an overview of our method’s Inter-Model Adapter. The Inter-Model Adapter is structured from three modules: the Image Feature Conversion Module, the Action Feature Conversion Module, and the Features Fusion Module. Below, we detail the components of each module and outline the processing procedure.

Image Feature Conversion Module. The Inter-Model Adapter of VideoChatGPT converting image features into tokens is used for this module. This process starts by applying both temporal and spatial mean pooling to the image features, $V_f \in \mathbb{R}^{T \times N \times D_f}$, extracted from each frame by the image language model. This process results in temporal features, $V_t \in \mathbb{R}^{T \times D_f}$, and spatial features, $V_n \in \mathbb{R}^{N \times D_f}$. Subsequently, these features are concatenated and then mapped to the LLM’s token space through a single linear layer, f_f , resulting in the converted image feature tokens, $Q_f = f_f([V_t, V_n]) \in \mathbb{R}^{(T+N) \times D_h}$. Here, the notation $[a, b]$ signifies the concatenation of vectors a and b .

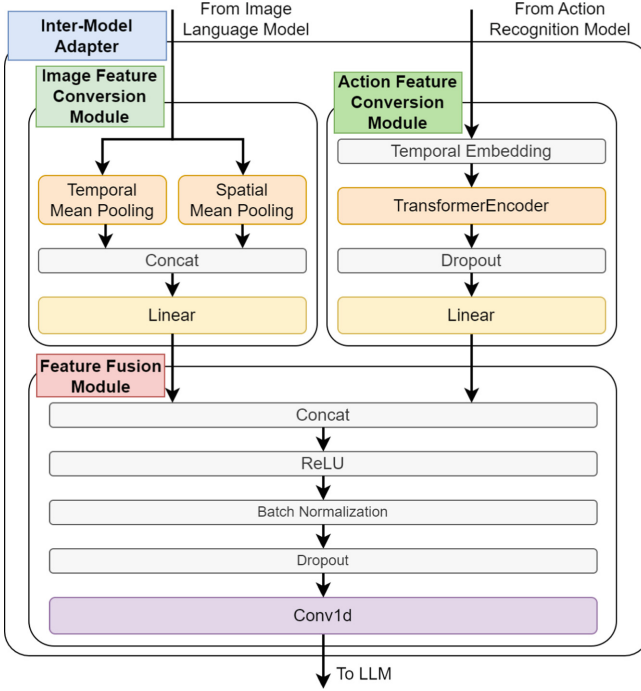


Fig. 2. The overview of Inter-Model Adapter.

Action Feature Conversion Module. This module is designed to analyze the interplay among action features within video segments and to map these features into the LLM’s token space effectively. The first function of this module is to capture global features that cannot be captured by segment-by-segment feature extraction by modeling the features in the temporal direction. To achieve this, it incorporates time embedding and a TransformerEncoder, with the TransformerEncoder set to a single layer featuring two heads mechanisms. Also, a single linear layer is utilized to map these analyzed features into the LLM’s token space. During the conversion of action features into action feature tokens in this module, the process starts with adding temporal embedding to the action features extracted per video segment by the action recognition model through the TransformerEncoder. This step produces an enhanced set of action features, $V'_s = \text{TransformerEncoder}(V_s + TE) \in \mathbb{R}^{T \times D_s}$, reflecting the temporal relationships between segments. Here, TE represents the temporal embedding that is the positional encoding in the temporal direction. Finally, a Dropout layer followed by a single linear layer f_s is applied, projecting the refined action features V'_s into the LLM’s token space, resulting in converted action feature tokens $Q_s = f_s(\text{Dropout}(V'_s)) \in \mathbb{R}^{T \times D_h}$.

Features Fusion Module. To merge the two distinct sets of features effectively, this module utilizes a one-dimensional convolution with a kernel size of one. The process starts by concatenating the image feature tokens, Q_f , and the action feature tokens, Q_s , from the feature conversion modules. This concatenated set then is processed by sequentially adapting ReLU, Batch Normalization, Dropout, and finally, the 1D convolution, resulting in the combined visual token, $Q_v = (\text{Conv1d}(\text{Dropout}(\text{BN}(\text{ReLU}([Q_f, Q_s]))) \in \mathbb{R}^{(2T+N) \times D_h}$, merged visual information of image and action information tailored for the LLM.

3.4 Data Augmentation

To address the challenge of insufficient training data our proposed method incorporates data augmentation techniques applied to the Video Instruction Dataset utilized for training. This augmentation process involves rephrasing existing instruction response texts, executed with the aid of Vicuna v1.5 [3] 13B. Specifically, paraphrases of the instructions are generated by instructing Vicuna to use synonyms and thesauruses extensively, avoid incorporating external information, and ensure the paraphrased instructions remain faithful to the original instruction-response relationship. This preserves the relationship between the instructions provided and the response, and extends the dataset without significantly deteriorating data quality.

3.5 Training

Our training approach follows Vision Instruction Tuning, utilizing a dataset comprised of video and corresponding instruction response text pairs, similar to Video-ChatGPT. The training objective is to minimize the token-by-token cross-entropy error between the actual responses and the model’s predictions.

The training process is divided into two distinct stages. In the first stage, only one visual encoder is active, and the feature conversion module corresponding is trained independently. The model structure at this stage of training is shown in Fig. 3a and Fig. 3b. This stage’s model architecture, when training the Image Feature Conversion Module, is similar to Video-ChatGPT [19], with the Image Feature Conversion Module being initialized using the inter-model adapter of Video-ChatGPT. The weights of the model-to-model adapter of Video-ChatGPT are equivalent to the weights of the Image Feature Conversion Module of the proposed method initialized with LLaVA [5] and then trained with the architecture shown in Fig. 3a using the non-augmented Video Instruction Dataset. Subsequently, in the second stage, both feature conversion modules are initialized with the weights trained in the first stage, and the entire Inter-Model Adapter, including the features fusion module, then are trained.

3.6 Prompts

The prompts for the LLM are crafted following the format established by Video-ChatGPT, structured as follows:

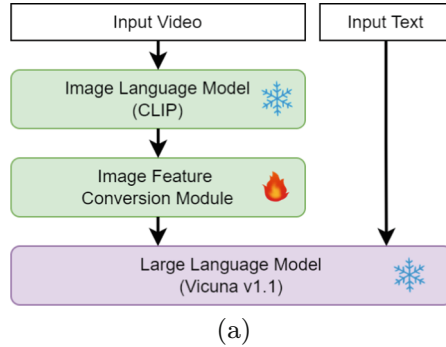


Fig. 3. (a) The model structure when training only Image Feature Conversion Module. (b) The model structure when training only Action Feature Conversion Module.

USER: ⟨Instruction⟩ ⟨Video-token⟩ ASSISTANT:

Here, ⟨Instruction⟩ denotes the instructions to the LLM, such as queries about the video, while ⟨Video-token⟩ symbolizes the visual features converted to tokens. The designations, “USER: and ASSISTANT:”, distinguish between user instructions and LLM responses, facilitating the LLM’s comprehension of dialogue progression, particularly in extended conversations. In our method, ⟨Instruction⟩ within the template is replaced by the actual instruction text and tokenized. Subsequently, the token for ⟨Video-token⟩ is substituted with the visual token Q_v , obtained by the Inter-Model Adapter, before being fed into the LLM.

4 Experiments

4.1 Experimental Settings

In our experiments, we follow the sampling parameters of Video-ChatGPT [19], setting the number of frames and video segments, T , to 100. The Dropout layer’s probability parameter, p , was adjusted to 0.0 during the first training stage and increased to 0.5 in the second stage. Additionally, the temperature parameter, τ , pivotal in controlling the probability distribution of LLM’s token generation during inference and thus influencing the model’s creativity, was fixed at 0.2, except where specified otherwise.

The training for both stages utilizes the same dataset and settings, employing the Video Instruction Dataset [19] derived from a subset of the ActivityNet dataset [6]. This dataset contains around 100,000 video pairs coupled with single-turn instruction-response texts. It is created by making instruction-response texts pertinent to video content using GPT-3.5 from human-crafted captions being included in a subset of ActivityNet dataset and frame-level captions from BLIP-2 [12]. As mentioned above, to address the scarcity of training data, our approach includes a data augmentation strategy, rephrasing instructions via a

Table 1. The number of questions in each category

Question Set	Action	Object	Total
GENERIC	1466	530	1996
TEMPORAL	481	18	499
CONSISTENCY	231	268	499

LLM, unlike Video-ChatGPT. Optimization is conducted using AdamW, with a learning rate schedule using linear warmup with a warmup rate of 0.03 and cosine decay with a peak at 2×10^{-5} . Each training stage is trained for three epochs, following the training of the inter-model adapter in Video-ChatGPT.

The quantitative evaluation is carried out by Video-based Generative Performance Benchmarking [19] and AutoEval-Video [23]. For the Video-based Generative Performance Benchmarking, a test set based on a subset of ActivityNet dataset [6] as well as the Video Instruction Dataset is used. The evaluation of each response is performed by GPT-3.5 (the checkpoints used is gpt-3.5-turbo-0125) to score a relative score on 0 to 5, based on comparison with the correct answers, in terms of perspective assigned to each data from five perspectives that are Correctness of Information (CI), Detail Orientation (DO), Contextual Understanding (CU), Temporal Understanding (TU) and Consistency (C). In the following, all questions are evaluated three times, and the means and standard deviations are reported for each item, except where specified otherwise. In addition, the evaluation questions in the Generative Performance Benchmarking dataset is divided into two types of questions using GPT-4o (the checkpoints used are gpt-4o-2024-05-13): action-oriented questions and object-oriented questions. Action-oriented questions mean the questions on dynamics in the videos where action features are expected to help to answer, while object-oriented questions mean the questions on objects and scenes for which image features are expected to be helpful. The number of both types are shown in Table 1. Note that, GENERIC is a split of the dataset used evaluating Correctness of Information, Detail Orientation, and Contextual Understanding. TEMPORAL is a split of the dataset used evaluating Temporal understanding, and CONSISTENCY is a split of the dataset used evaluating Consistency. The evaluation results for each of the two types of questions are reported as well.

For the AutoEval-Video, a uniquely collected and annotated dataset for the benchmark from YouTube across multiple capability domains and topics is used. The evaluation of each response was performed by GPT-4 (the checkpoints used are gpt-4-1106-preview) to judge right and wrong based on the specific evaluation rules defined for each sample, in terms of the perspective assigned to each sample from nine perspectives: Dynamic Perception, State Transition Perception, Comparison Reasoning, Reasoning with External Knowledge, Explanatory Reasoning, Predictive Reasoning, Description, Counterfactual Reasoning and Camera Movement Perception. In the following, the means of the accuracy of overall and each item of three times evaluations conducted are reported.

In our study, emphasis was placed on the results of Video-based Generative Performance Benchmarking, as this is the most commonly used method in existing Video-LLM assessments. The results of AutoEval-Video, on the other hand, were used to check the generalisation performance of the model, as they were based on dataset collected and annotated in a completely different way to the training data.

4.2 Comparison with Baseline

A quantitative comparative analysis by Video-based Generative Performance Benchmarking between our proposed method and Video-ChatGPT [19], Video-LLaMA [4] is shown in Table 2. Table 2 also includes the results of the evaluation of responses by GPT-4o for reference. GPT-4o generated the responses based on the following instruction, using 20 frames sampled from the video: “These images are frames cut from a single video. Referring to these images, answer the following questions. However, the actual answers do not require frame-by-frame explanations, please generate the actual answer to the aggregated video”. Note that the evaluation of GPT-4o was conducted only once. To make fair comparison, we show the results excluding data augmentation (denoted as w/o data aug.) and the results training inter-model adapters with only augmented Video Instruction Dataset without pre-training with such as LLaVA [5] dataset (denoted as scratch). Also, the results for action-oriented questions and object-oriented questions are shown in Table 3 and Table 4.

Our method superior performance across all metrics when compared to existing models. Also, within the same metrics, there is no significant difference in standard deviations between different methods. Notably, even in the absence of data augmentation, our approach surpassed Video-ChatGPT in all but Consistency. This underscored the significant impact of integrating action features on enhancing the response performance of Video-LLM responses. Note that our method is still clearly inferior to the response by GPT-4o, indicating room for further development of the open source Video-LLM.

On the other hand, Table 3 and Table 4 show that Act-ChatGPT outperforms Video-ChatGPT, especially for action-oriented questions, while conversely the performance of Act-ChatGPT is slightly less than the baseline in the evaluations for only object-oriented questions. Therefore, our proposed method can be regarded as focusing on action-oriented questions more.

In addition, a quantitative comparative analysis by AutoEval-Video between our proposed method and Video-ChatGPT, Video-LLaMA, is detailed in Table 5 and Table 6. As with the Video-based benchmark, we show the results excluding data augmentation (denoted as w/o data aug.) and the results training inter-model adapters with only augmented Video Instruction Dataset (denoted as scratch). In this evaluation, by contrast, our method underperformed the base model on almost all items. Thus, it can be said that our proposed method has poorer generalization performance than the Video-ChatGPT. The poor performance of Act-ChatGPT for AutoEval-Video mainly comes from the differences with and without pre-training of inter-model adapters.

Table 2. Results of Video-based Generative Performance Benchmarking.

	CI \uparrow	DO \uparrow	CU \uparrow	TU \uparrow	C \uparrow
Video-LLaMA	2.23 \pm 1.25	2.16 \pm 0.79	2.52 \pm 1.13	1.93 \pm 1.09	2.02 \pm 1.09
Video-ChatGPT	2.50 \pm 1.33	2.31 \pm 0.85	2.87 \pm 1.18	2.10 \pm 1.15	2.20 \pm 1.24
Video-ChatGPT (scratch)	2.44 \pm 1.31	2.29 \pm 0.83	2.82 \pm 1.17	2.10 \pm 1.11	2.06 \pm 1.19
Act-ChatGPT (scratch)	2.53 \pm 1.34	2.33 \pm 0.82	2.89 \pm 1.21	2.19 \pm 1.15	2.17 \pm 1.23
Act-ChatGPT (w/o data aug.)	2.53 \pm 1.36	2.33 \pm 0.86	2.92 \pm 1.19	2.13 \pm 1.14	2.17 \pm 1.23
Act-ChatGPT	2.62 \pm 1.35	2.37 \pm 0.85	3.00 \pm 1.17	2.20 \pm 1.14	2.28 \pm 1.25
GPT-4o	4.02 \pm 1.13	3.46 \pm 0.92	4.19 \pm 0.92	3.30 \pm 1.30	3.54 \pm 1.23

Table 3. Results of Video-based Generative Performance Benchmarking for the action-oriented questions.

	CI \uparrow	DO \uparrow	CU \uparrow	TU \uparrow	C \uparrow
Video-LLaMA	2.16 \pm 1.11	2.08 \pm 0.68	2.41 \pm 1.04	1.90 \pm 1.05	2.13 \pm 1.14
Video-ChatGPT	2.51 \pm 1.23	2.25 \pm 0.78	2.85 \pm 1.13	2.09 \pm 1.12	2.49 \pm 1.24
Video-ChatGPT (scratch)	2.50 \pm 1.22	2.28 \pm 0.78	2.86 \pm 1.13	2.10 \pm 1.07	2.43 \pm 1.21
Act-ChatGPT (scratch)	2.65 \pm 1.23	2.35 \pm 0.76	3.00 \pm 1.15	2.20 \pm 1.14	2.60 \pm 1.24
Act-ChatGPT (w/o data aug.)	2.62 \pm 1.25	2.32 \pm 0.80	2.97 \pm 1.14	2.13 \pm 1.12	2.54 \pm 1.24
Act-ChatGPT	2.72 \pm 1.24	2.36 \pm 0.78	3.08 \pm 1.11	2.19 \pm 1.10	2.72 \pm 1.23

The inter-model adapter of Video-ChatGPT is pre-trained with 753k LLaVA [5] training images and fine-tuned with non-augmented 100k Video Instruction Dataset, whereas the one for Act-ChatGPT is trained with only augmented 200k Video Instruction Dataset from scratch, except for the Image Feature Conversion Module, which is initialized with the weights of the inter-model adapter of Video-ChatGPT. This means that the Image Feature Conversion Module in the Act-ChatGPT was pre-trained with 753k LLaVA training images as well, whereas the action feature conversion module was not pre-trained with any dataset. This is due to the fact that the proposed method uses a segment-based action recognition model for one of the visual encoders in which not image data but video data is used for training. In fact, when comparing Video-ChatGPT (scratch) and Act-ChatGPT (scratch) from Table 5, which were trained inter-model adapters only on the augmented Video Instruc-

Table 4. Results of Video-based Generative Performance Benchmarking for the object-oriented questions.

	CI \uparrow	DO \uparrow	CU \uparrow	TU \uparrow	C \uparrow
Video-LLaMA	2.43 \pm 1.56	2.39 \pm 1.00	2.81 \pm 1.29	2.61 \pm 1.76	1.92 \pm 1.03
Video-ChatGPT	2.49 \pm 1.58	2.48 \pm 1.01	2.90 \pm 1.29	2.26 \pm 1.89	1.94 \pm 1.19
Video-ChatGPT (scratch)	2.27 \pm 1.51	2.32 \pm 0.95	2.73 \pm 1.26	2.15 \pm 1.84	1.75 \pm 1.09
Act-ChatGPT (scratch)	2.20 \pm 1.56	2.27 \pm 0.96	2.60 \pm 1.32	1.83 \pm 1.53	1.80 \pm 1.10
Act-ChatGPT (w/o data aug.)	2.28 \pm 1.59	2.36 \pm 1.00	2.77 \pm 1.30	2.33 \pm 1.63	1.84 \pm 1.13
Act-ChatGPT	2.33 \pm 1.58	2.37 \pm 1.02	2.79 \pm 1.28	2.35 \pm 1.89	1.89 \pm 1.14

Table 5. Results of AutoEval-Video (overall)

	All↑
Video-LLaMA	0.070
Video-ChatGPT	0.101
Video-ChatGPT (scratch)	0.045
Act-ChatGPT (scratch)	0.049
Act-ChatGPT (w/o data aug.)	0.064
Act-ChatGPT	0.064

Table 6. Results of AutoEval-Video (each item)

	Dynamic ↑	State Transitions ↑	Comparison ↑
Video-LLaMA	0.059	0.073	0.140
Video-ChatGPT	0.088	0.115	0.246
Video-ChatGPT (scratch)	0.044	0.094	0.176
Act-ChatGPT (scratch)	0.050	0.041	0.123
Act-ChatGPT (w/o data aug.)	0.036	0.083	0.123
Act-ChatGPT	0.029	0.073	0.193
	External Knowledge ↑	Explanatory ↑	Predictive ↑
Video-LLaMA	0.084	0.040	0.041
Video-ChatGPT	0.084	0.086	0.135
Video-ChatGPT (scratch)	0.016	0.035	0.031
Act-ChatGPT (scratch)	0.042	0.045	0.000
Act-ChatGPT (w/o data aug.)	0.062	0.066	0.062
Act-ChatGPT	0.050	0.066	0.052
	Description ↑	Counterfactual ↑	Camera Movement ↑
Video-LLaMA	0.056	0.140	0.000
Video-ChatGPT	0.044	0.123	0.111
Video-ChatGPT (scratch)	0.022	0.035	0.111
Act-ChatGPT (scratch)	0.011	0.176	0.000
Act-ChatGPT (w/o data aug.)	0.067	0.053	0.000
Act-ChatGPT	0.044	0.123	0.000

tion Dataset without pre-training with image data, Act-ChatGPT outperforms Video-ChatGPT on overall accuracy. This shows that the performance deterioration in the evaluation with AutoEval-Video observed in Act-ChatGPT is due to the lack of pre-training data, not to the introduction of the proposed action features. This does not negate the effectiveness of the proposed method.

Figure 4 shows qualitative comparisons between Act-ChatGPT and Video-ChatGPT. The observations from the top and middle response results in Fig. 4 illustrate that Act-ChatGPT enhances responses over Video-ChatGPT by improving action recognition as well as the identification of objects involved in these actions. It is conceivable that this improvement in object recognition is due to the fact that the large language model recognizes action features in a different way to spatial features, allowing the consistency of object and action elements as sentences to be taken into account when generating responses. Furthermore, the



Fig. 4. Examples of responses.

bottom response result in Fig. 4 demonstrates that Act-ChatGPT retains the capability to recognize unique objects, as observed in Video-ChatGPT.

4.3 Ablation Studies

In the ablation studies, Video-based Generative Performance Benchmarking is used. Table 7 displays the quantitative results for Act-ChatGPT under various settings. Specifically, (w/o Stage1) denotes results from training solely in the second stage, (w/o Fusion) refers to scenario not using a features fusion module, while (w/o Image) and (w/o Action) refer to scenarios where only the action recognition model or the image language model is employed as a vision encoder, respectively. Note that the evaluation was conducted only once in each setting and the results are reported. As a side note, even when only one visual encoder is used, the feature fusion module was applied by adjusting the number of dimensions. The findings reveal a notable decline in performance metrics when Act-ChatGPT is trained solely during the second stage, underscoring the critical role of multi-stage learning. Moreover, utilizing only one type of visual encoder, whether for actions or images, leads to significant drops in all metrics. These outcomes suggest that image and action features play complementary roles in video understanding and emphasize the benefits of action features utilized in video understanding. In addition, focusing on feature fusion, the performance significantly deteriorated is found when features were not fused. This shows that in this research, where a LLM is frozen, a mechanism for explicitly fusing features is important for improving the performance.

Table 7. Results of Video-based Generative Performance Benchmarking under various settings.

	CI \uparrow	DO \uparrow	CU \uparrow	TU \uparrow	C \uparrow
Act-ChatGPT (w/o Stage1)	2.28 \pm 1.32	2.20 \pm 0.89	2.66 \pm 1.24	2.00 \pm 1.16	2.01 \pm 1.46
Act-ChatGPT (w/o Image)	2.17 \pm 1.34	2.03 \pm 0.85	2.46 \pm 1.22	1.86 \pm 1.07	1.99 \pm 1.12
Act-ChatGPT (w/o Action)	2.41 \pm 1.31	2.21 \pm 0.82	2.74 \pm 1.20	2.19 \pm 1.16	1.97 \pm 1.23
Act-ChatGPT (w/o Fusion)	2.39 \pm 1.32	2.23 \pm 0.89	2.74 \pm 1.24	2.12 \pm 1.16	2.26 \pm 1.21
Act-ChatGPT (w/ all)	2.62 \pm 1.35	2.37 \pm 0.85	3.00 \pm 1.17	2.20 \pm 1.14	2.28 \pm 1.25

5 Limitations

In our study, a new Act-ChatGPT with a newly introduced action feature was proposed. Several limitations still remain. The first major limitation relates to training data. In the recent trends in Video-LLMs, as Peng Jin *et al.* [9] have shown the advantages of joint learning of images and videos, it has become mainstream to learn various visual representations by also utilizing a large amount of image data in addition to video data. However, in our work, we used an action recognition model that operated on a video segment basis as part of the visual encoder. This design choice made it difficult to utilize image data for training. Therefore, to keep up with these trends and achieve better performance, it is necessary to create extensive video datasets to compensate for the lack of data or develop methods to utilize images for training.

The second limitation is in the computational cost. The computational cost of our proposed method is relatively high because it employs a large action recognition model. Additionally, our dual-encoder approach, which processes a certain amount of object features in the action branch, further contributes to these costs. The action recognition model used in our proposed method, trained on the Kinetics dataset [11] that are considered relatively easy to classify even with only scene information, includes somewhat object recognition capabilities. However, these capabilities are sometimes redundant in our method since the image branch already handles object recognition. This redundancy suggests a need for the more focused and compact model that extracts only movement features, which could help in reducing computational expenses.

6 Conclusions

In this paper, we proposed Act-ChatGPT, a Video-LLM designed to use action features from individual video segments to enrich response generation with insight into the action depicted. Act-ChatGPT enhanced both action and their associated object recognition capabilities, outperforming the Video-ChatGPT used as a base model. In addition, it also retains a certain level of object recognition capabilities, such as identifying unique objects in the video, demonstrating the improvement in video understanding over the base approaches overall.

Acknowledgments. This work was supported by JSPS KAKENHI Grant Numbers, 22H00540 and 22H00548.

References

1. Alec, R., Jong, Wook, K., et al.: Learning transferable visual models from natural language supervision. In: ICML, vol. 139, pp. 8748–8763 (2021)
2. Chenfei, W., Shengming, Y., Weizhen, Q., Xiaodong, W., Zecheng, T., Nan, D.: Visual ChatGPT: talking, drawing and editing with visual foundation models. [arXiv:2303.04671](https://arxiv.org/abs/2303.04671) (2023)
3. Chiang, W.L., Li, Z., et al.: Vicuna: an open-source chatbot impressing GPT-4 with 90%* ChatGPT quality (2023). <https://lmsys.org/blog/2023-03-30-vicuna/>
4. Hang, Z., Xin, L., Lidong, B.: Video-LLaMA: an instruction-tuned audio-visual language model for video understanding. [arXiv:2306.02858](https://arxiv.org/abs/2306.02858) (2023)
5. Haotian, L., Chunyuan, L., Qingyang, W., Yong, Jae, L.: Visual instruction tuning. In: NeurIPS (2023)
6. Heilbron, F.C., Escorcia, V., Ghanem, B., Niebles, J.C.: ActivityNet: a large-scale video benchmark for human activity understanding. In: CVPR, pp. 961–970 (2015)
7. Hugo, T., Louis, M., et al.: LLaMA 2: open foundation and fine-tuned chat models. [arXiv:2307.09288](https://arxiv.org/abs/2307.09288) (2023)
8. Hugo, T., et al.: LLaMA: open and efficient foundation language models. [arXiv:2302.13971](https://arxiv.org/abs/2302.13971) (2023)
9. Jin, P., Takanobu, R., Zhang, C., Cao, X., Yuan, L.: Chat-UniVi: unified visual representation empowers large language models with image and video understanding. [arXiv:2311.08046](https://arxiv.org/abs/2311.08046) (2023)

10. Kaplan, J., et al.: Scaling laws for neural language models. [arXiv:2001.08361](#) (2020)
11. Kay, W., et al.: The kinetics human action video dataset. [arXiv:1705.06950](#) (2017)
12. Li, J., Li, D., Savarese, S., Hoi, S.: BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In: ICML, pp. 19730–19742 (2023)
13. Li, K., et al.: VideoChat: chat-centric video understanding. [arXiv:2305.06355](#) (2023)
14. Li, K., et al.: MVBench: a comprehensive multi-modal video understanding benchmark. [arXiv:2311.17005](#) (2023)
15. Li, K., et al.: UniFormerV2: unlocking the potential of image ViTs for video understanding. In: ICCV, pp. 1632–1643 (2023)
16. Li, K., et al.: Unmasked Teacher: towards training-efficient video foundation models. In: ICCV, pp. 19948–19960 (2023)
17. Li, Y., Wang, C., Jia, J.: LLaMA-VID: an image is worth 2 tokens in large language models. [arXiv:2311.17043](#) (2023)
18. Lin, B., et al.: Video-LLaVA: learning united visual representation by alignment before projection. [arXiv:2311.10122](#) (2023)
19. Muhammad, M., Hanoona, R., Salman, K., Fahad, Shahbaz, K.: Video-ChatGPT: towards detailed video understanding via large vision and language models. [arXiv:2306.05424](#) (2023)
20. Vaswani, A., et al.: Attention is all you need. In: NeurIPS, vol. 30 (2017)
21. Wang, L., et al.: VideoMAE V2: scaling video masked autoencoders with dual masking. In: CVPR, pp. 14549–14560 (2023)
22. Wei, J., et al.: Finetuned language models are zero-shot learners. In: ICLR (2022)
23. Xiuyuan, C., Yuan, L., Yuchen, Z., Weiran, H.: AutoEval-video: an automatic benchmark for assessing large vision language models in open-ended video question answering. [arXiv:2311.14906](#) (2023)
24. Zhao, W.X., Zhou, K., et al.: A survey of large language models. [arXiv:2303.18223](#) (2023)
25. Zhu, B., et al.: LanguageBind: extending video-language pretraining to n-modality by language-based semantic alignment. [arXiv:2310.01852](#) (2023)
26. Zoph, B., et al.: Emergent abilities of large language models. In: Proceedings of Transactions on Machine Learning Research (2022)



Machine Vision-Aware Quality Metrics for Compressed Image and Video Assessment

Mikhail Dremin¹(✉) , Konstantin Kozhemyakov¹ , Ivan Molodetskikh¹ ,
Malakhov Kirill², Sagitov Artur^{2,3}, and Dmitriy Vatolin¹ 

¹ Lomonosov Moscow State University, Moscow, Russia
{mikhail.dremin,konstantin.kozhemiakov,ivan.molodetskikh,
dmitriy}@graphics.cs.msu.ru

² Huawei Technologies Co., Ltd., Shenzhen, China

³ Shenzhen, China

Abstract. A main goal in developing video-compression algorithms is to enhance human-perceived visual quality while maintaining file size. But modern video-analysis efforts such as detection and recognition, which are integral to video surveillance and autonomous vehicles, involve so much data that they necessitate machine-vision processing with minimal human intervention. In such cases, the video codec must be optimized for machine vision. This paper explores the effects of compression on detection and recognition algorithms (objects, faces, and license plates) and introduces novel full-reference image/video-quality metrics for each task, tailored to machine vision. Experimental results indicate our proposed metrics correlate better with the machine-vision results for the respective tasks than do existing image/video-quality metrics.

Keywords: Machine vision · Image Quality · Video Compression · Object Detection · Face Recognition · License Plate Recognition

1 Introduction

As the field of computer-vision continues to evolve, an increasing number of algorithms are being deployed in real-world applications. A popular application of this technology is video analytics, which has become integral to video surveillance, autonomous vehicles and other systems. Video analytics, for example, has two crucial tasks: detection and recognition. Depending on the system, the subjects of these tasks can be traffic signs, vehicles (object detection), human faces (detection/recognition), license plates (detection/recognition), and so on. As the number of video-surveillance cameras increases, automating these tasks becomes more critical given that human operators are incapable of processing such vast quantities of data.

S. Artur—Independent Researcher Linjianping.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2025
A. Antonacopoulos et al. (Eds.): ICPR 2024, LNCS 15323, pp. 266–282, 2025.
https://doi.org/10.1007/978-3-031-78347-0_18

To ensure efficient storage and transmission of such extensive data, the captured images and videos require compression. Lossy compression standards such as JPEG, H.264/AVC, H.265/HEVC, and AV1 serve this purpose; their development involved optimizing the visual quality of the compressed content. The best way to assess visual quality is subjective human ratings, but they can be time-consuming and expensive. Hence the use of full reference (FR) quality-assessment methods such as PSNR, SSIM [24], and VMAF [2], some of which correlate highly with subjective scores [5]. These methods enable us to quickly and cost-effectively configure and develop codecs while emphasizing visual quality.

Most state-of-the-art detection and recognition algorithms are based on deep neural networks, and their effectiveness is evaluated not visually, but through performance metrics. Compression directly affects the performance and reliability of computer-vision algorithms, especially at high compression ratios [6, 19]. Vision researchers and developers have therefore attempted to determine image quality for algorithms and develop codecs for machine vision [1].

For video surveillance system the main objects for analysis are people and vehicles, thus we choose three main video analytics algorithms for our machine-oriented quality metrics: object detection (including vehicles, persons, faces, and license plates), face recognition, license plates recognition.

Often video surveillance systems use a specific detection/recognition algorithm (e.g. YOLOv5). Optimizing camera-data compression for them requires a comparatively fast method that predicts detection/recognition performance on the already encoded video, thereby enabling selection of encoding parameters to maximize that performance. If the target neural-network-based detection/recognition algorithms are evaluating the relative performance of codec prototypes or codec settings, they need lots of time and computational power owing to the number of parameters and the neural-network size. For instance, the recent x264 codec has almost 50 settings; selection of these parameters through an exhaustive search, even without using complex neural networks, would take centuries [30].

During investigation for our machine oriented metric we have following targets:

1. Achieve high correlation score with mentioned three main video analytics algorithms for its particular implementations with lower computational complexity.
2. Considering question about metric generalization for different implementation detection/recognition algorithms.

Additionally, detection/recognition algorithms are imperfect, and identifying the cause of potential errors is impossible. For example, an object could be truly unrecognizable in the encoded image or video, necessitating quality improvement, or one algorithm may have certain limitations whereas another can detect the object error-free. Running multiple detection/recognition algorithms to improve the robustness of such an evaluation method would be even more time-consuming and computationally intensive.

To address these issues, our paper makes three important contributions:

1. First, we propose a methodology of measuring image and video quality in terms of detection/recognition performance.
2. Second, we analyze detection- and recognition-performance correlation with that of popular image-quality-assessment (IQA) and video-quality-assessment (VQA) methods on widely used image and video codecs. Our results show little to no correlation between their outputs and detection/recognition performance.
3. Third, we propose new video-quality metrics based on convolutional-neural-network (CNN) models with respect to object detection, face detection/recognition, and license-plate detection/recognition performance. We validated our metrics by checking their correlation with the performance of the machine-vision algorithms for the corresponding tasks.

2 Related Work

Methods for image-quality assessment have garnered considerable attention from researchers in the field of visual-perception as high-quality video content is important for retaining viewer interest [16]. Initially, these efforts evaluated image quality on the basis of how the human eye perceives visual information in generic videos and in specific video types such as streaming, user-generated content (UGC), 3D, and virtual and augmented reality. Recently, although computer vision increasingly permeates everyday life, efforts to develop machine-vision-aware image- and video-quality metrics have been less extensive.

2.1 Objective Image and Video-Quality Assessment

Many methods and algorithms evaluate the visual quality of images. Among the most widely used are PSNR and SSIM; they assess image quality on the basis of signal (pixel) similarity between the original and evaluated images. When applied to videos, these algorithms work frame by frame and then average the results. More-modern approaches to assessing video quality have emerged: for example, VMAF demonstrates a higher correlation with subjective human evaluations relative to PSNR and SSIM.

There are also no reference (NR) methods that take only a single image as input. Among them is an early NR quality metric: NIQE [17], which evaluates an image’s “naturalness” and serves in cases where the original image is unavailable. Recently, NR metrics have approached the quality of FR metrics. For instance, DOVER [25] and MDTVSFA [13] correlate highly with subjective quality [5].

These methods are widely used to develop and optimize video-compression and video-processing algorithms. Some exhibit high correlation with human-perceived visual quality, but they were not designed to predict detection and recognition performance on compressed images and video.

2.2 Image and Video-Quality Assessment for Detection

Kong et al. [12] introduced a no-reference IQA algorithm for object detection. This algorithm integrates classical computer vision and selects 13 image features, including gradient-vector metrics and HOG descriptors [9]. Random forest is trained to predict detection quality using the revised frame-detection-accuracy metric, which is akin to Intersection over Union (IoU).

Rahman et al. [20] presented an algorithm that employs statistical features based on the internal representations of images input to a neural-network-based object detector. It then uses these statistics to train a LightGBM algorithm, which performs classification to predict whether the object-detection accuracy for a given input image will surpass a certain threshold or fail.

Schubert et al. [22] suggested predicting an object-detection algorithm's accuracy on the basis of its confidence in the results. The authors analyzed non-maximum suppression step using features and statistics from the detection results to predict accuracy without ground-truth annotations. Their underlying hypothesis is that the more objects this stage filters out, the higher the confidence in the remaining object's actual presence.

Beniwal et al. [7] proposed a metric based on the quantization error from H.264 compression. The mean DCT ratio of all filters of first Faster R-CNN convolutional layer is used as a quality label. Higher values indicate higher quantization loss and lower quality. They train CNN network to predict these quality labels using cropped patches as an input.

2.3 Image and Video-Quality Assessment for Recognition

Best-Rowden and Jain [8] introduced an automatic method for predicting the quality of face images, integrating two assessment strategies: subjective evaluations by humans and objective measurement based on similarity scores for face recognition utility. Both approaches utilize features from deep neural networks as inputs for SVMs, allowing for a quantification of face image quality that reflects human perception and the operational performance of face-recognition systems.

Hernandez-Ortega et al. [11] introduced FaceQnet, a tool that estimates the quality of face images with respect to their utility in face recognition. FaceQnet operates by fine-tuning a preexisting face-recognition neural network for regression; the goal is to predict face-image quality as a continuous value.

Terhorst et al. [23] proposed an unsupervised approach to face-image-quality estimation called SER-FIQ. They compute the quality score as the mean Euclidean distance of the multiple embedding features from a recognition model with different dropout patterns.

Ou et al. [18] introduced SDD-FIQA, a method that uses inter- and intraclass comparison scores to determine the quality of face images by creating distributions of genuine and impostor scores for each image. The quality of an image is assessed by calculating the mean Wasserstein distance between these distributions across multiple iterations. This approach then refines a face-recognition model with these quality scores, similarly to the FaceQNet method.

Most of these studies consider quality loss due to shooting conditions-such as poor lighting, motion blur, and noise-but neglect artifacts that arise during compression with different quality factors. They also use knowledge of architecture and intermediate results of detection/recognition algorithms, although in some cases the algorithm is inaccessible.

3 Datasets

Our research hinges on carefully collected datasets, each of which is pivotal to developing and testing image-quality metrics. We used the validation and/or test parts of popular datasets pertinent to the respective tasks: COCO 2017 [14] for object detection, WIDER FACE [27] for face detection, CCPD [26] for license-plate detection and recognition, and CelebA [15] for face recognition. Our test set contained proprietary unlabeled videos from CCTV cameras for all tasks, except Glint360k images for face recognition.

To ensure our method’s proper function despite various distortions, we selected several practical video and image codecs (rav1e, x264, x265, VVenC, and JPEG) to encode the dataset images. We balanced the dataset by calculating the PSNR of the compressed images and adjusting the codec quality parameters to achieve a similar distribution over all codecs, as Fig. 1 illustrates.

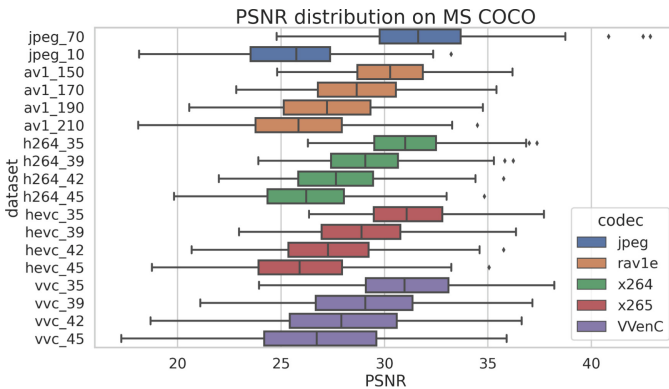


Fig. 1. PSNR distribution on MS COCO dataset, compressed with different codec and quality factors.

In total, every image in our datasets has 20 quality degradations. For JPEG, we used 20 compression degrees but retained only two quality factors because they represent two boundary values: the minimum, below which severe image degradation occurs, and the maximum, above which quality improvement is indiscernible. Note that we were unable to obtain completely uncompressed data; it would have been helpful, however, as open datasets usually employ JPEG compression and our proprietary videos employ H.264/H.265, so our compression distortions are on top of existing compression. This limitation should not significantly affect the

results because open datasets have undergone quality control and avoid extreme compression, and because the camera codecs have a high bitrate.

Test-Dataset Labeling. The videos in our test set were unlabeled, but ground-truth labels are crucial for correct target-metric calculations, correlation-score measurement, and object extraction from images. We therefore used an automatic labeling pipeline in four of five tasks: object detection, face detection, license-plate detection, and license-plate recognition.

We extracted all the frames from videos and ran detection algorithms to label frames (each case used the algorithm’s most complex version for precise labeling: i.e., YOLOv5X for object detection, RetinaFace for face recognition, and LPRNet for license-plate recognition). We selected only objects with a corresponding high confidence and ignored detector errors. Instead, we looked at performance deterioration on compressed videos and picked about 1,000 frames containing several objects per task. Our choices were approximately 100 frames apart—since nearby frames are usually similar - to ensure diversity and reduce the number of noisy labels. We employed distinct frames (images), not videos, to train and validate results because video detection and recognition usually consider each frame and average the results for the entire video. Figure 2 show labeling examples. For license-plate detection we applied an extra filter using a recognition algorithm: it picked frames with fully recognized characters and filtered frames with similar license plates using the Levenshtein distance. The datasets were subsequently reviewed by humans for gross detection errors (false positive and false negative detections). The pixel-level accuracy of the bounding boxes was not meticulously verified: even though automatic annotation is often inaccurate, it is reasonable to expect that if a sophisticated detection algorithm for GT labeling fails to identify an object in an uncompressed image, the target detection algorithm will also struggle to detect the object once the image undergoes compression.



Fig. 2. Detection labeling example.

The face recognition test set was derived from the open Glint360k [3] dataset due to the need to extract at least two different face images for the same person. This requirement arose because our video recordings featured very similar facial images for each person, as individuals seldom appeared more than once in

external video-surveillance footage. For each person in a dataset we choose two face images: one for the database and one for a query.

For the database we attempt to find the person’s “best” face image using ICAO-compliance software (Biolab-ICAO, as in FaceQNet’s pipeline). For a query we select another random image of that person’s face. Figure 3 shows example images. Table 1 summarizes the characteristics of our study’s datasets.



Fig. 3. Face images examples.

Table 1. Summary of datasets used in the study.

Task	Dataset part	Source images	Compressed images
Object detection	Train and val sets, COCO	3,125	62,500
	Test set, proprietary	1,000	20,000
Face detection	Train and val sets, WIDER	3,226	64,520
	Test set, proprietary	1,000	20,000
Car plate detection	Train and val sets, CCPD	5,020	100,400
	Test set, proprietary	600	12,000
Face recognition	Train and val sets, CelebA	5,000	100,000
	Test set, Glint360k	1,000	20,000
Car plate recognition	Train and val sets, CCPD	5,020	100,400
	Test set, proprietary	600	12,000

4 Proposed Method

4.1 Detection Methodology

Object detection, crucial for applications like surveillance, identifies and locates objects in images by combining classification and localization. Key performance metrics include confidence score and Intersection over Union (IoU) with ground truth (GT). These metrics are essential for distinguishing between false negatives (missed detections) and false positives (incorrect detections), especially when objects are scarce. Correctly identifying missed detections is particularly important in surveillance, as human oversight can address false positives.

Object detectors tend to underperform in images that contain small or occluded objects [21], a limitation attributable to their implementation details

rather than compression effects. Given our focus on information loss due to compression, it is essential to select quality metrics that are not biased by object-detector limitations.

The Mean Average Precision (mAP) metric assesses the detection algorithm's confidence and incorporates an IoU threshold to determine the object-matching accuracy. However, its utility diminishes in images that contain few objects: here, mAP values may provide no meaningful insight because they may be binary (e.g., 0 or 1 for images with a single object).

Similarly, Average Precision (AP), which accounts for precision and recall, is poorly suited to single images or frames. The metric fails to differentiate between the effects of information loss (manifesting as false negatives) and detector inaccuracies (leading to false positives), particularly under varied compression levels.

We consider three possible target-detection performance metrics: mean-IoU, Object IoU, and Delta Object IoU. Mean-IoU provides a general measure of how accurately objects are detected over the entire image, regardless of object number or size. It permits consideration of false negatives: if no correct match to a reference object is found, the IoU for that object is zero. Furthermore, it allows setting of an IoU-value threshold below which objects are no longer considered correctly detected. Note, however, that although mean-IoU facilitates evaluation of individual images, it does not directly reflect the total number of objects, potentially obscuring detection performance relative to object quantity.

Object IoU focuses on the IoU for an object cropped from a reference frame. It provides a more specific measure of detection accuracy for individual objects by assessing how well the detected object matches the GT in size and location. Object IoU is particularly useful for analyzing detection performance object by object, a potentially critical capability for applications that must precisely detect each object.

Delta Object IoU represents the difference in an object's IoUs between the reference frame and the compressed frame. Essentially, it quantifies the impact of compression on individual-object detection quality. A small Delta Object IoU indicates that the object's detection accuracy is relatively unaffected by compression, whereas a large Delta Object IoU suggests considerable detection-performance degradation due to compression.

We then investigated which of our proposed target metrics is most representative for evaluating detection performance.

4.2 Detection Metric

All target detectors in this study are YOLOv5 variations tailored to specific detection tasks: YOLOv5s for object detection, YOLOv5Face for face detection, and LPD YOLOv5 for license-plate detection. First we analyzed almost 40 IQA and VQA metrics to determine how they correlate with the detectors' performance. We applied each metric to every compressed image in our datasets.

Our analysis included the following:

- Image/video-quality metrics
 - Full-reference: PSNR (peak signal-to-noise ratio), SSIM (structural similarity index), MS-SSIM (multiscale structural similarity index), VMAF (video multi-method assessment fusion)
 - No-reference: NIQE (natural image quality evaluator), BRISQUE (Blind/Referenceless image spatial quality evaluator)
- Other Metrics: SAM (spectral angle mapper), SRE (spatial resolution enhancement), DSS (decision support system), NLPD (normalized Laplacian pyramid distance), GMSD (gradient magnitude similarity deviation), MDSI (mean deviation similarity index), VSI (visual saliency-induced Index), ERQA (edge-based region quality assessment)
- NSS (natural scene statistics): blockiness, total variation, colorfulness, brightness

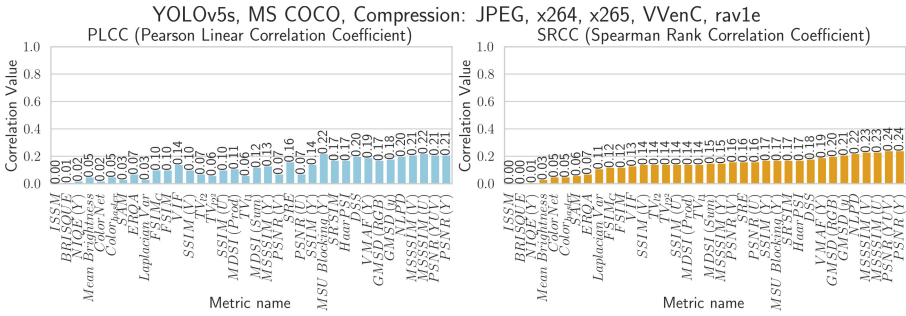


Fig. 4. Object detection. Objective metric results

After verifying the comparison method, we found the correlation scores for all tested metrics to be low, approximately 0.2–0.3 according to SRCC (see Fig. 4, Table 2). This result suggests none of the metrics are practical for evaluating detection performance on compressed videos.

Table 2. Obtained correlations for standart IQA/VQA quality metrics.

Method			Best IQA/VQA metric results among tested	
Task	Dataset	Target algorithm	PLCC	SRCC
Object Detection	COCO	YOLOv5s	0.21	0.24
Face Detection	WIDER	YOLO5Face	0.25	0.33
Car plate Detection	CCPD	LPD YOLOv5	0.32	0.31

Figure 5 illustrates our proposed metric’s overall architecture. The input varies with the target metric: for mean-IoU, we used whole images; for object-IoU and Delta Object IoU, we used cropped objects based on GT bounding boxes (Fig. 6 shows this variant).

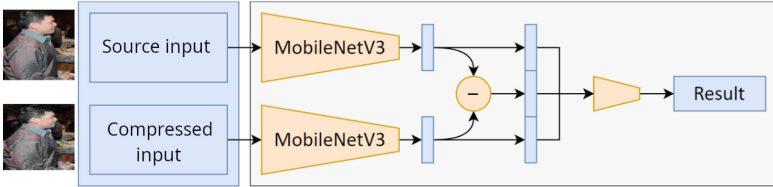


Fig. 5. The proposed quality metric architecture.

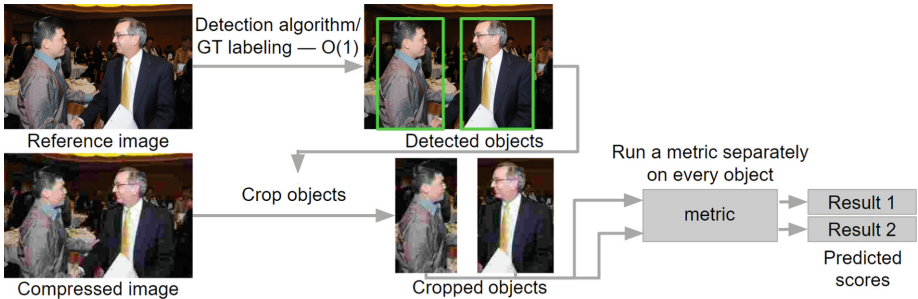


Fig. 6. The metric pipeline for assessing object crops.

Due to IQA and VQA metrics’ low correlation, we decided to research novel video-quality metrics based on CNN models which process crops of input images.

The lightweight MobileNetV3 network served as backbone for feature extraction. We then concatenated embeddings with their difference and fed the results to an MLP block that predicts the target metric.

We sequentially examined each proposed target-detection-performance metric to determine which was most representative. The first was mean-IoU. We trained our metric to predict the mean-IoU score for the entire compressed image as an input and achieved low SRCC scores of 0.29–0.37, depending on the task. Consequently, our analysis identified a major drawback of mean-IoU as the target performance metric. The IoU for each detected object in an image can vary, and averaging the results can lead to a false detection-quality perception in compressed content: the average value heavily masks omission of objects (false negatives). This drawback is absent from the next two metrics, which average the IoU of all objects in the dataset.

Having already calculated bounding boxes in uncompressed sequences (GT), we examined object IoU and computed the average value for cropped objects

across all images. This approach recognizes that vast areas of a video frame, such as trees, asphalt, and sidewalks, are typically irrelevant to surveillance. By concentrating on areas that contain objects of interest, we sought to assess how the quality of these areas degrades rather than evaluating overall image or background quality. After training our metric to predict Object IoU as the target, we achieved an SRCC of 0.5–0.6 for detection. The results showed that the metric poorly predicted absolute IoU values. Indeed, predicting an IoU score for an object without any supporting information from the detector is difficult. Rather than considering compression’s impact on detection-algorithm performance, Object IoU considers the trained metric’s performance to predict the detection result, leading to misalignment with the actual objective.

To address described limitations, we shifted to the Delta Object IoU. That shift acknowledges the inherent detection-performance variability among compression levels and seeks to more directly quantify compression’s impact. Our experimental results demonstrate a notable improvement in correlation scores when using Delta Object IoU. Specifically, we observed SRCC in the 0.8–0.9 range for multiple tasks, indicating a stronger correlation with detection-algorithm performance degradation due to compression. Delta Object IoU’s main disadvantage as a target metric is that the deltas of different detection algorithms have low correlation, so the trained algorithm generalizes poorly to different target detectors.

Because of the unified approach of our proposed detection metrics, we can aggregate them into a general version. This version, trained on all datasets for all tasks, yielded results that were slightly inferior to those we obtained for task-specific metrics (see Table 3).

Table 3. Correlation scores (SRCC) for the general metric compared to individual metrics.

Dataset (task)	Test data	
	Generalized model	Separate models
All datasets	0.837	–
Object detection	0.844	0.892
Face detection	0.816	0.818
Car plate detection	0.811	0.811

4.3 Face-Recognition Methodology

Standard metrics for measuring face-recognition efficiency include false-acceptance rate (FAR) and false-rejection rate (FRR). However, these can only be calculated for an entire dataset, whereas sometimes it is necessary to assess the quality of a single photo. A common methodology for neural-network-based face recognition involves computing embeddings for a corpus of reference images, each

associated with known individuals, as well as for a query image. The identification process entails choosing the reference image whose embedding is most similar to that of the query image. To quantify the impact of image compression on face recognition performance, we calculated ArcFace [10] embeddings for all images (database and queries). Consider cosine similarity between image embeddings:

$$R(I_a, I_b) = \text{cosSim}(\text{emb}(I_a), \text{emb}(I_b)) \quad (1)$$

The proposed target metric is calculated as follows:

$$F(I_{ref}, I_{compr}) = R(I_{ref}, I_{database}) - R(I_{compr}, I_{database}) \quad (2)$$

Compression algorithms will influence correct recognition of the query face, so the cosine similarity between database and compressed images will differ. The face-recognition performance metric measures this difference in cosine similarity; if the difference is high, the face-recognition system is more likely to fail.

4.4 Face-Recognition Metric

Recognition often follows detection, so we focused on image crops. As with detection, our first task was to analyze the performance of standard IQA/VQA metrics and existing quality-assessment methods for face recognition. The correlation scores for all metrics were low, suggesting that none is practical for evaluating face-recognition performance on compressed videos (Fig. 7).

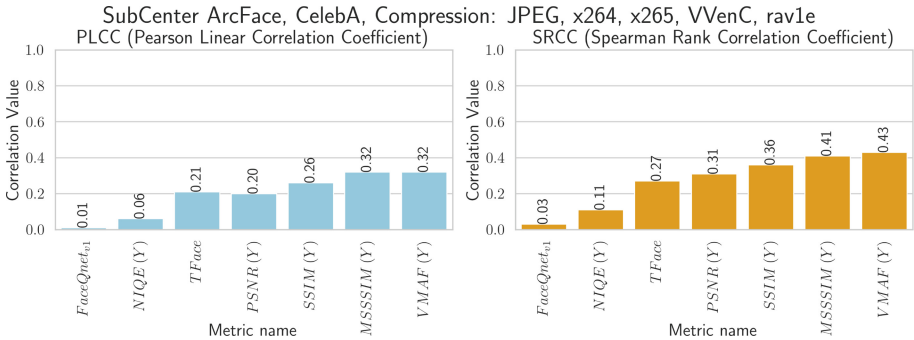


Fig. 7. Existing quality assessment methods for face recognition and standard IQA/VQA metrics results.

For face recognition we used ResNet-18 and a one-layer regression head to predict cosine-similarity deltas between source and compressed images. First, we trained our metric to predict a target score for a pair of reference and compressed face crops; the result was a 0.59 SRCC. We thus concluded that the

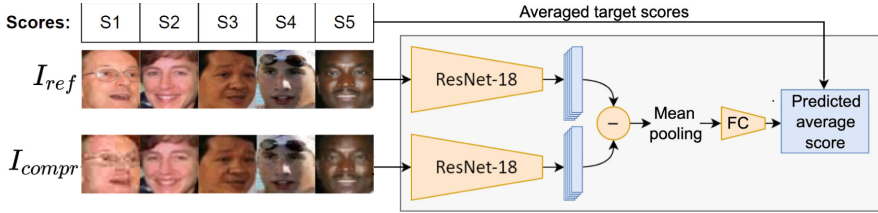


Fig. 8. Pipeline of face-recognition metric.

metric provides unstable results on individual face images. To address this instability, we propose predicting image/video quality on small subsets of images as single inputs and averaging CNN feature before regression head to predict scores (Fig. 8). The SRCC of the metric trained with the proposed strategy is 0.85 SRCC.

4.5 License-Plate-Recognition Methodology

To crop license plates from the full-size images, we used bounding boxes from GT. In license-plate recognition, the outcome is a sequence of characters, so the quality assessment differs fundamentally from other tasks considered. A variety of string-similarity measures, such as Hamming distance and Levenshtein distance, calculate the minimum number of character transformations necessary to convert one string to another. But these measures are unnormalized, making them less convenient for regression. Additionally, the Hamming distance requires equal-length strings, whereas license-plate-recognition methods may output a varying number of characters owing to unreadable symbols or misinterpretations, such as compression artifacts.

For this work, we propose Jaro similarity, which considers string lengths as well as the number and placement of common characters, making it suitable for automatic comparison and classification in data matching, duplicate identification, and more. The Jaro similarity d_j of two given strings s_1 and s_2 is

$$d_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3} \left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m} \right) & \text{otherwise} \end{cases}$$

where: m is the number of *matching characters*; t is half the number of *transpositions*.

It is particularly useful for license-plate recognition, as it accounts for matching characters that may be shifted a few positions left or right. Similar to IoU, the overall metric for an image comes from averaging Jaro similarity over all matched license plates.

4.6 License-Plate-Recognition Metric

The standard IQA/VQA methods exhibit bad correlation scores for our proposed target metric (Fig. 9).

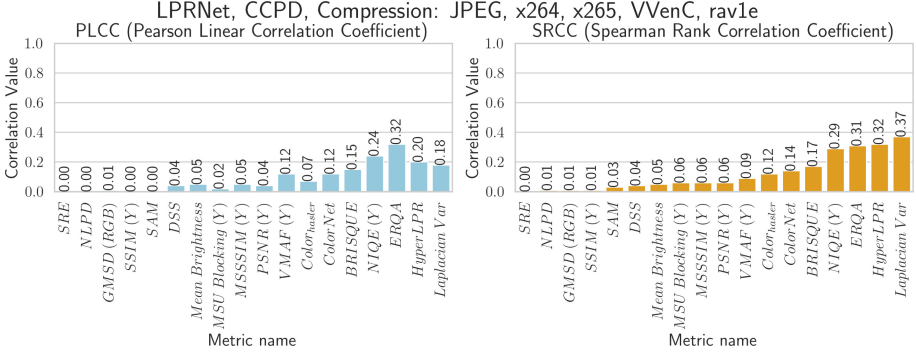


Fig. 9. Standard IQA/VQA metrics results.

We developed a neural-network metric to predict the performance of a license-plate-recognition method that takes as input a grayscale license-plate image. The model is a classic CNN, and its output is a single real number representing the expected recognition quality (Fig. 10). A value of 1 indicates perfect recognition, while a value of 0 indicates the license-plate characters will be incorrectly recognized. After training the architecture, we achieved an SRCC of 0.85.

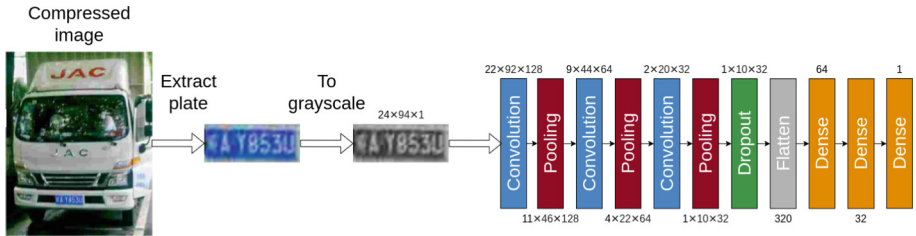


Fig. 10. Pipeline of license-plate-recognition metric.

5 Conclusion

This paper comprehensively explores the development and validation of novel image-quality metrics for machine vision, with a focus on assessing detection and recognition performance on compressed images and videos. These metrics address a major gap in standard IQA/VQA methods, which cater mainly to

human visual perception and inadequately predict machine-vision performance on compressed images and videos.

Our metrics proved to be 3–5 times more computationally efficient than the target examined algorithms. For detection, we produced a general metric that applies to various detection subtasks, making the computations even more efficient since several detectors of the same family (e.g., YOLOv5) are used simultaneously in some cases (e.g., object and license-plate detection).

Although our proposed metrics yielded promising results, they are designed for specific tasks and, unmodified, may be inapplicable to other machine-vision tasks. Future work could explore development of more-general metrics that can handle a wider range of detection and recognition tasks. Additionally, further research could examine integration of these metrics into video-compression algorithms to automate optimization for machine vision.

Our metrics can also be used for other tasks that consume image/video data as an input. For example, visual question answering uses multimodal fusion of features extracted from both image and text. Taking a look at the existing VideoQA models, we can observe that the majority of them are utilizing features (rather than predicted labels) extracted from an image or a video by familiar object detection models, e.g. Faster-RCNN features in ViteVQA [29] and BUTD [4], or ResNet features in MFH [28]. Therefore, we expect that our results will extrapolate to those models. Applicability to newer models (e.g. Fuyu-8B) that use direct linear projection of image patches requires a separate study. As our machine-vision metrics are feature-focused for image/video data, they can serve as a reference point for developing a metric for the VisualQA task.

Acknowledgements. This study was supported by Russian Science Foundation under grant 24-21-00172, <https://rscf.ru/en/project/24-21-00172/>.

References

1. Standards - MPEG-AI. <https://www.mpeg.org/standards/MPEG-AI/>
2. Toward a practical perceptual video quality metric. <https://netflixtechblog.com/toward-a-practical-perceptual-video-quality-metric-653f208b9652>
3. An, X., et al.: Partial FC: training 10 million identities on a single machine. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 1445–1449 (2021)
4. Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., Zhang, L.: 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)
5. Antsiferova, A., Lavrushkin, S., Smirnov, M., Gushchin, A., Vatolin, D., Kulikov, D.: Video compression dataset and benchmark of learning-based video-quality metrics. *Adv. Neural. Inf. Process. Syst.* **35**, 13814–13825 (2022)
6. Aqqa, M., Mantini, P., Shah, S.K.: Understanding how video quality affects object detection algorithms. In: VISIGRAPP (5: VISAPP), pp. 96–104 (2019)
7. Beniwal, P., Mantini, P., Shah, S.K.: Image quality assessment using deep features for object detection. In: VISIGRAPP (4: VISAPP), pp. 706–714 (2022)

8. Best-Rowden, L., Jain, A.K.: Learning face image quality from human assessments. *IEEE Trans. Inf. Forensics Secur.* **13**(12), 3064–3077 (2018)
9. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), vol. 1, pp. 886–893. IEEE (2005)
10. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2019)
11. Hernandez-Ortega, J., Galbally, J., Fierrez, J., Haraksim, R., Beslay, L.: FaceQnet: quality assessment for face recognition based on deep learning. In: 2019 International Conference on Biometrics (ICB), pp. 1–8. IEEE (2019)
12. Kong, L., Ikusan, A., Dai, R., Zhu, J.: Blind image quality prediction for object detection. In: 2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR), pp. 216–221. IEEE (2019)
13. Li, D., Jiang, T., Jiang, M.: Unified quality assessment of in-the-wild videos with mixed datasets training. *Int. J. Comput. Vision* **129**(4), 1238–1257 (2021)
14. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
15. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3730–3738 (2015)
16. Min, X., Duan, H., Sun, W., Zhu, Y., Zhai, G.: Perceptual video quality assessment: a survey. arXiv preprint [arXiv:2402.03413](https://arxiv.org/abs/2402.03413) (2024)
17. Mittal, A., Soundararajan, R., Bovik, A.C.: Making a “completely blind” image quality analyzer. *IEEE Signal Process. Lett.* **20**(3), 209–212 (2012)
18. Ou, F.Z., et al.: SDD-FIQA: unsupervised face image quality assessment with similarity distribution distance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 7670–7679 (2021)
19. O’Byrne, M., Sugrue, M., Kokaram, A., et al.: Impact of video compression on the performance of object detection systems for surveillance applications. In: 2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 1–8. IEEE (2022)
20. Rahman, Q.M., Sunderhauf, N., Dayoub, F.: Per-frame map prediction for continuous performance monitoring of object detection during deployment. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 152–160 (2021)
21. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. *Adv. Neural Inf. Process. Syst.* **28** (2015)
22. Schubert, M., Kahl, K., Rottmann, M.: MetaDetect: uncertainty quantification and prediction quality estimates for object detection. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–10. IEEE (2021)
23. Terhorst, P., Kolf, J.N., Damer, N., Kirchbuchner, F., Kuijper, A.: SER-FIQ: unsupervised estimation of face image quality based on stochastic embedding robustness. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5651–5660 (2020)
24. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.* **13**(4), 600–612 (2004)

25. Wu, H., et al.: Exploring video quality assessment on user generated contents from aesthetic and technical perspectives. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 20144–20154 (2023)
26. Xu, Z., Yang, W., Meng, A., Lu, N., Huang, H.: Towards end-to-end license plate detection and recognition: a large dataset and baseline. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 255–271 (2018)
27. Yang, S., Luo, P., Loy, C.C., Tang, X.: Wider face: a face detection benchmark. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5525–5533 (2016)
28. 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.* **29**(12), 5947–5959 (2018). <https://doi.org/10.1109/tnnls.2018.2817340>
29. Zhao, M., et al.: Towards video text visual question answering: benchmark and baseline. *Adv. Neural. Inf. Process. Syst.* **35**, 35549–35562 (2022)
30. Zvezdakov, S., Solovyov, A., Vatolin, D.: Iterative machine-learning-based method of selecting encoder parameters for speed-bitrate tradeoff. In: 2022 Data Compression Conference (DCC), p. 01. IEEE (2022)

Author Index

A

Artur, Sagitov 266

B

Bai, Mengchao 82

C

Castillo, Myrna 114

Clément, Michaël 67

Contractor, Sohail 51

D

Dahaghin, Mahtab 114

Dai, Xin 36

De Ridder, Dirk 179

Del Bue, Alessio 114

Deng, Haoge 36

Deng, Jeremiah D. 179

Devarakota, Pandu 98

Dremin, Mikhail 266

E

El-Baz, Ayman 51

F

Ferrigno, Luigi 192

G

Gala, Apurva 98

Ghazal, Mohammed 51

Giraud, Rémi 67

Githinji, P. Bilha 82

J

Jiang, Haoran 147

K

Kato, Zoltan 131

Kenmochi, Yukiko 1

Khelifi, Adel 51

Kirill, Malakhov 266

Kozhemyakov, Konstantin 266

Kurtz, Camille 19

L

Li, Ke 36

Li, Ming 233

Liang, Wen 82

Lin, Yuke 233

Liu, Dong 233

Liu, Xiao 82

M

Mahmoud, Ali 51

Manning, Patrick 179

Mantini, Pranav 98

Mendes Forte, Julien 1

Mirza, Samiha 98

Molinara, Mario 192

Molodetskikh, Ivan 266

Mustafa, Hamza 192

N

Naegel, Benoît 19

Nakamizo, Yuto 250

Nguyen, Vuong D. 98

O

Osa, Priscilla Indira 131

P

Passat, Nicolas 1, 19

Perrin, Romain 19

Q

Qi, Yonggang 36

Qin, Peiwu 82

R

Ran, Peipei 82
Randrianasoa, Jimmy Francky 19

S

Sandhan, Tushar 204
Shah, Shishir K. 98
Sharafeldeen, Ahmed 51
Su, Jianbo 219
Sun, Jing 219

T

Toso, Matteo 114
Tyagi, Divyani 204

V

Vatolin, Dmitriy 266
Vitelli, Michele 192

W

Wang, Yang 147
Wei, Xiaoge 163
Wu, Qiang 147
Wu, Yanan 147

X

Xu, Jingwen 163
Xu, Wenbo 147

Y

Yaghi, Maha 51
Yanai, Keiji 250
Yue, Yuan 179
Yuen, Pong C. 163

Z

Zerubia, Josiane 131
Zhang, Jian 147
Zhao, Wei 82
Zhou, Huali 233