

Li Ning
Vincent Chau
Francis Lau (Eds.)

Communications in Computer and Information Science

1362

Parallel Architectures, Algorithms and Programming

11th International Symposium, PAAP 2020
Shenzhen, China, December 28–30, 2020
Proceedings

Editorial Board Members

Joaquim Filipe 

Polytechnic Institute of Setúbal, Setúbal, Portugal

Ashish Ghosh

Indian Statistical Institute, Kolkata, India

Raquel Oliveira Prates 

Federal University of Minas Gerais (UFMG), Belo Horizonte, Brazil

Lizhu Zhou

Tsinghua University, Beijing, China


More information about this series at <http://www.springer.com/series/7899>


Li Ning · Vincent Chau ·
Francis Lau (Eds.)

Parallel Architectures, Algorithms and Programming

11th International Symposium, PAAP 2020
Shenzhen, China, December 28–30, 2020
Proceedings

Editors

Li Ning 
Shenzhen Institutes of Advanced
Technology, Chinese Academy of Sciences
Shenzhen, China

Vincent Chau 
Shenzhen Institutes of Advanced
Technology, Chinese Academy of Sciences
Shenzhen, China

Francis Lau 
The University of Hong Kong
Hong Kong, China

ISSN 1865-0929 ISSN 1865-0937 (electronic)
Communications in Computer and Information Science
ISBN 978-981-16-0009-8 ISBN 978-981-16-0010-4 (eBook)
<https://doi.org/10.1007/978-981-16-0010-4>

© Springer Nature Singapore Pte Ltd. 2021

This work is subject to copyright. All rights are reserved 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 Singapore Pte Ltd. The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

Welcome to PAAP 2020, the 11th International Symposium on Parallel Architectures, Algorithms and Programming. PAAP is an annual international conference for scientists and engineers in academia and industry to present their research results and development activities in all aspects of parallel architectures, algorithms, and programming techniques.

Following the successful PAAP 2008 in Hefei, PAAP 2009 in Nanning, PAAP 2010 in Dalian, PAAP 2011 in Tianjin, PAAP 2012 in Taipei, PAAP 2014 in Beijing, PAAP 2015 in Nanjing, PAAP 2017 in Haikou, PAAP 2018 in Taiwan, and PAAP 2019 in Guangzhou, PAAP 2020 will be held in Shenzhen, China.

This year we received 75 submissions, and we selected 37 papers. The submissions were in general of high quality, making paper selection a tough task. The paper review process involved all Program Committee members. To ensure a high-quality program and provide sufficient feedback to authors, we made a great effort to have each paper reviewed by three independent reviewers on average.

It would not have been possible for PAAP 2020 to take place without the help and support of various people. The efforts of the authors, Program Committee members, and reviewers were essential to the conference's quality, and all deserve our utmost appreciation. We also wish to thank the Local Organization Committee members for all their hard work in making PAAP 2020 a great success, and thank Communications in Computer and Information Science (CCIS) of Springer for their support.

We hope that all participants enjoyed PAAP 2020 and had a great time in Shenzhen.

December 2020

Li Ning
Vincent Chau
Francis Lau

Local Arrangements Committee

Yuyang Li	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Fumin Qi	National Supercomputing Centre in Shenzhen, China

Program Committee

Costin Bădică	University of Craiova, Romania
Guangdong Bai	The University of Queensland, Australia
Vincent Chau	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Yawen Chen	University of Otago, New Zealand
Chunru Dong	Hebei University, China
Yichuan Dong	National Supercomputing Center in Shenzhen, China
Zisen Fang	National Supercomputing Center in Shenzhen, China
Guichen Gao	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Huaxi Gu	Xidian University, China
Longkun Guo	Qilu University of Technology, China
Ajay Gupta	Western Michigan University, US
Lu Han	Academy of Mathematics and Systems Science, Chinese Academy of Sciences, China
Xinxin Han	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Ping He	Hebei University of Economics and Business, China
Qiang Hua	Hebei University, China
Zhiyi Huang	University of Otago, New Zealand
Haiping Huang	Nanjing University of Posts and Telecommunications, China
Mirjana Ivanović	University of Novi Sad, Serbia
Graham Kirby	University of St Andrews, United Kingdom
Kai-Cheung Leung	University of Auckland, New Zealand
Kenli Li	Hunan University, China
Yamin Li	Hosei University, Japan
Shuangjuan Li	South China Agricultural University, China
Min Li	Shandong Normal University, China
Shangsong Liang	Sun Yat-sen University, China
Zhongzhi Luan	Beihang University, China
Marin Lujak	IMT Lille Douai, France
Li Ning	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Cheng Qiao	University College Cork, Ireland
Yingpeng Sang	Sun Yat-sen University, China
Neetesh Saxena	Cardiff University, United Kingdom
Guang Tan	Sun Yat-sen University, China

Haisheng Tan	University of Science and Technology of China, China
Jingjing Tan	Weifang University, China
Lei Wang	Peking University, China
Xin Wang	Fudan University, China
Yinling Wang	Dalian University of Technology, China
Di Wu	Sun Yat-sen University, China
Weigang Wu	Sun Yat-sen University, China
Zijun Wu	Hefei University, China
Chenchen Wu	Tianjin University of Technology, China
Jun Wu	Beijing Jiaotong University, China
Jigang Wu	Guangdong University of Technology, China
Qiang Xu	Wenzhou University Oujian College, China
Yicheng Xu	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Ruiqi Yang	University of Chinese Academy of Sciences, China
Xinfeng Ye	University of Auckland, New Zealand
Jingjing Yu	Beijing Jiaotong University, China
Dongxiao Yu	Shandong University, China
Filip Zavoral	Charles University, Czech Republic
Feng Zhang	Hebei University, China
Dongmei Zhang	Shandong Jianzhu University, China
Zonghua Zhang	Huawei France, France
Xiaoyan Zhang	Nanjing Normal University, China
Yu Zhang	University of Science and Technology of China, China
Yong Zhang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Zhenning Zhang	Beijing University of Technology, China
Haibo Zhang	University of Otago, New Zealand
Cheng Zhong	Guangxi University, China
Chunyue Zhou	Beijing Jiaotong University, China
Rong Zhou	Shenzhen Institutes of Advanced Technology, Institutes of Advanced Technology, China
Yifei Zou	The University of Hong Kong, China

Contents

Restoring Survivable Spanning Tree: An Alternative Algorithm	1
<i>Lulu Zheng, Yalan Wu, Jiale Huang, Peng Liu, and Jigang Wu</i>	
On the Decycling Problem in a Torus	12
<i>Antoine Bossard</i>	
Deep Learning Optimization for Many-Core Virtual Platforms	22
<i>Hengyu Cai, Chengming Ning, and Qilong Zheng</i>	
Development of Low-Cost Indoor Positioning Using Mobile-UWB-Anchor-Configuration Approach	34
<i>Ang Liu, Shiwei Lin, Xiaoying Kong, Jack Wang, Gengfa Fang, and Yunlong Han</i>	
BERT-CoQAC: BERT-Based Conversational Question Answering in Context	47
<i>Munazza Zaib, Dai Hoang Tran, Subhash Sagar, Adnan Mahmood, Wei E. Zhang, and Quan Z. Sheng</i>	
Streaming Algorithms for Monotone DR-Submodular Maximization Under a Knapsack Constraint on the Integer Lattice	58
<i>Jingjing Tan, Dongmei Zhang, Hongyang Zhang, and Zhenning Zhang</i>	
NASIL: Neural Network Architecture Searching for Incremental Learning in Image Classification	68
<i>Xianya Fu, Wenrui Li, Qiurui Chen, Lianyi Zhang, Kai Yang, Duzheng Qing, and Rui Wang</i>	
Nonsubmodular Maximization with Knapsack Constraint via Multilinear Extension.	81
<i>Jiachen Ju, Min Li, Jianxin Liu, Qian Liu, and Yang Zhou</i>	
FEENET: A Real-Time Semantic Segmentation via Feature Extraction and Enhancement	93
<i>Sixiang Tan, Wenzhong Yang, JianZhuang Lin, and Weijie Yu</i>	
Parallel Reconstruction for High Resolution Multi-frame Solar Speckle Images.	104
<i>Lingxiao Zhu, Murong Jiang, Pengming Fu, and Wenhao Chui</i>	
Analysing and Forecasting Electricity Demand and Price Using Deep Learning Model During the COVID-19 Pandemic.	115
<i>Israt Fatema, Xiaoying Kong, and Gengfa Fang</i>	

Cross-database Micro Expression Recognition Based on Apex Frame Optical Flow and Multi-head Self-attention.	128
<i>Jiebin Wen, Wenzhong Yang, LieJun Wang, Wenyu Wei, Sixiang Tan, and Yongzhi Wu</i>	
GPS Intelligent Solution of Aerial Image Target in State Grid EIA Survey. . .	140
<i>Yu Wu, Fei Wang, Caihua Sun, Songyang Zhang, Jie Huang, Zhentao Liu, and Wei Sun</i>	
Encryption and Decryption in Conic Curves Cryptosystem Over Finite Field $GF(2^n)$ Using Tile Self-assembly	150
<i>Yongnan Li</i>	
Optimizing Embedding-Related Quantum Annealing Parameters for Reducing Hardware Bias.	162
<i>Aaron Barbosa, Elijah Pelofske, Georg Hahn, and Hristo N. Djidjev</i>	
A Behavioural Network Traffic Novelty Detection for the Internet of Things Infrastructures	174
<i>Salma Abdalla Hamad, Quan Z. Sheng, Dai Hoang Tran, Wei Emma Zhang, and Surya Nepal</i>	
A Fast Algorithm for Image Segmentation Based on Global Cosine Fitting Energy Model.	187
<i>Yanping Chen, Le Zou, Zhize Wu, Qianjing Huang, and Xiaofeng Wang</i>	
Household Garbage Classification: A Transfer Learning Based Method and a Benchmark	200
<i>Qian Cheng, Zhi-Ze Wu, Zi-Jun Wu, Le Zou, Huan-Yi Li, and Xiao-Feng Wang</i>	
Lightweight Neural Network Based Garbage Image Classification Using a Deep Mutual Learning	212
<i>Xiao Liu, Zhi-Ze Wu, Zi-Jun Wu, Le Zou, Li-Xiang Xu, and Xiao-Feng Wang</i>	
VBSSR: Variable Bitrate Encoded Video Streaming with Super-Resolution on HPC Education Platform	224
<i>Run Wu, Gangqiang Zhou, Miao Hu, and Di Wu</i>	
An Investigation on the Performance of Highly Congested Home WiFi Networks During the COVID-19 Pandemic	236
<i>Rhys Cunningham, Ben Hendry, and Wee Lum Tan</i>	
Using Feed-Forward Network for Fast Arbitrary Style Transfer with Contextual Loss.	247
<i>Peixiang Chen, Yujie Zhang, Jiangyi Huang, and Zhanghui Liu</i>	

Enhancing Underwater Image Using Multi-scale Generative Adversarial Networks	259
<i>Yujie Zhang, Peixiang Chen, Jiangyi Huang, and Yuzhong Chen</i>	
Inferring Prerequisite Relationships Among Learning Resources for HPC Education	270
<i>Run Wu, Chenyu Luo, Miao Hu, and Di Wu</i>	
Research on Bank Knowledge Transaction Coverage Model Based on Innovation Capacity Analysis	282
<i>Ming Zhu, Zhenyu Wang, Xiangyang Feng, Pengyu Wan, Wenpei Shao, and Ran Tao</i>	
Deep Deterministic Policy Gradient Based Resource Allocation in Internet of Vehicles.	295
<i>Zhuo Ma, Xin Chen, Teng Ma, and Ying Chen</i>	
A Pufferfish Privacy Mechanism for the Trajectory Clustering Task	307
<i>Yuying Zeng, Yingpeng Sang, Shunchao Luo, and Mingyang Song</i>	
A Novel Attention Model of Deep Learning in Image Classification	318
<i>Qiang Hua, Liyou Chen, Pan Li, Shipeng Zhao, and Yan Li</i>	
FDRA: Fully Distributed Routing Architecture for Private Virtual Network in Public Cloud.	331
<i>Zhangfeng Hu, Hui Zhang, Siqing Sun, Chuanji Gao, Yanjun Li, and Xiong Li</i>	
Energy Consumption Modeling for the PageRank Application in Spark	343
<i>Yunlan Wang, Yingjing Wang, Zhengxiong Hou, and Xingli Li</i>	
Distributed Dense Tucker Decomposition Based on Hierarchical SVD.	355
<i>Zisen Fang, Fumin Qi, Yichuan Dong, Yong Zhang, and Shengzhong Feng</i>	
A Multi-objective Task Offloading Strategy for Workflow Applications in Mobile Edge-Cloud Computing.	365
<i>Yongqiang Gao and Dandan Yan</i>	
A Deep Reinforcement Learning Based Feature Selector	378
<i>Yiran Cheng, Kazuhiko Komatsu, Masayuki Sato, and Hiroaki Kobayashi</i>	
Automatic Thread Block Size Selection Strategy in GPU Parallel Code Generation	390
<i>Weifang Hu, Lin Han, Pu Han, and Jiandong Shang</i>	

**SPORTS: A Semi-partitioned Real-Time Scheduler for Heterogeneous
Multicore Platforms 405**
Yanshu Sharma, Zinea Das, and Sanjay Moulik

**Boosting Performance in Parallel Computing Models with a New
Experimental Architecture 418**
*Alberto Arteta Albert, Akshay Harshakumar,
Luis Fernando de Mingo López, and Nuria Gómez Blas*

**RRW: A Reliable Ring Waveguide-Based Optical Router for Photonic
Network-on-Chips 429**
Meaad Fadhel, Lei Huang, and Huaxi Gu

Author Index 439



Restoring Survivable Spanning Tree: An Alternative Algorithm

Lulu Zheng, Yalan Wu, Jiale Huang, Peng Liu, and Jigang Wu^(✉)

School of Computers, Guangdong University of Technology, Guangzhou, China
asjgwucn@outlook.com

Abstract. In the modern network with high data transmission rate, coping with the link failures rapidly becomes a major network fault-tolerant challenge. Existing techniques adopt a survivable spanning connection (SSC) including two not-fully-disjoint spanning trees to prevent link failures. However, the spanning trees in SSC will be invalid when the shared links fail. This paper aims to algorithmic technique for fast restoring the invalid spanning trees in SSC. Firstly, we model an optimization problem of restoring the SSC based on the existing invalid spanning trees. Then restoration algorithm is proposed for the faulty shared link by two spanning trees. In the proposed algorithm, the edges linking the two subtrees of one tree with the fault are collected to form a set of the candidate links. Then, the link with the minimum failure probability is selected from the candidate set, and it is added into the spanning trees for generating a new SSC. Experimental results show that the recovery time is reduced up to 36% in comparison to the latest work cited in this paper, for the network with sizes varying from 10 to 100. Meanwhile, the loss rate of survivability is kept no more than 1%.

Keywords: Survivability · Survivable spanning connection · Spanning tree · Faulty link · Fault-tolerance

1 Introduction

Nowadays the data rate of Ethernet is 100 Gb/s and beyond, achieving almost 400 Gb/s in the near future [1]. With the rapid growth of network data transmission rates, the failure of network infrastructure definitely poses a great threat to the data transmission. Each link has a failure probability in the network, however, the link will fail due to other reasons, such as natural disasters, intentional attacks [2] or sudden traffic bursts [3]. The fast restoration techniques are proposed for the faulty link, which is important for the reduction of the data loss.

Supported in part by project of Guangdong Science and Technology Plan under Grant 2019B010118001, and the National Natural Science Foundation of China under Grant 61871475.

The network survivability is the ability of a network to recover data transmission from failure. It is important to design the effective survivability schemes for the protection of data disruption due to link failures. Most existing works improve the network survivability by designing the fast and robust network recovery schemes on the *single link failure model* [4]. Especially, the robust survivability schemes must consider quality of service (QoS) metrics, such as recovery time, data loss, capacity requirements, and time complexity of the recovery algorithm [5]. For reducing the data loss dramatically, the recovery time should be no more than 50 ms for the faulty link of the network [6].

A *spanning tree* contains all nodes and part links of the network, and it is widely used to broadcast messages with the minimum overhead of data transmission in a variety of networks. The protection schemes using backup spanning trees are less overhead than using backup paths. The link-disjoint spanning trees are studied, which have no shared links between backup spanning trees [7, 8], and used for the reliable algorithms in different networks [9]. The protection schemes with full link-disjoint spanning trees have a survivability level of 1. However, there is excessive redundancy. The concept of *tunable survivability* is introduced in [10], which provides a quantitative measure to specify the desired level of survivability. That is, the survivability of network can be supported in the range of 0% to 100% by using not-fully-disjoint spanning trees. In general, the protection scheme with more spanning trees can increase the survivability level but it also imposes higher management overhead. Therefore, it is important to calculate the number of not-fully-disjoint trees with optimal survivability and reasonable communication costs. Yallouz *et al.* [4] verified that two not-fully-disjoint survivable spanning trees (T_1 and T_2) can guarantee a nearly optimal survivability. That is, there are some shared links between T_1 and T_2 . The protection scheme with T_1 and T_2 is called a survivable spanning connection (SSC).

Currently, the single protection scheme, such as the survivable spanning connection, only provides a backup spanning tree for the faulty link. The backup spanning tree also fails and the data transmission is interrupted when the shared link fails between the T_1 and T_2 . For this case, the protection schemes have to regenerate two survivable spanning trees and form a new survivable spanning connection. However, it takes more computing time, especially in a large network.

Summarizing the above-mentioned protection schemes, we have observed the following challenges. 1) The protection scheme should be enabled to deal with any dynamic link failure. 2) The protection scheme with the faulty link should be quickly restored. 3) The survivability of the protection scheme should be close to the maximum survivability of the current network.

This paper focuses on the restoration for the spanning trees in the survivable spanning connection, which can be restored quickly by adjusting and updating the invalid spanning trees, instead of the traditional restoration schemes by regenerating new spanning trees. Moreover, our approach enables to deal with dynamic link failure in real-time, and results in the survivable spanning

connection with the acceptable survivability. The main contributions of this paper are summarized as follows.

- The optimization problem of restoring the SSC on the basis of the existing invalid spanning trees is formalized, instead of discarding the invalid spanning trees and regenerating spanning trees as did as in the existing work cited in this paper.
- The fast restoration algorithm is proposed for the formalized problem. The algorithm is to deal with the failure of shared links by T_1 and T_2 in the SSC, to restore both spanning trees simultaneously. The algorithm directly adjusts the original SSC, by updating T_1 and T_2 to T'_1 and T'_2 , respectively, to generate a new survivable connection.
- The proposed algorithm is simulated and compared with the latest work in [10]. The simulation results show that our algorithm can significantly reduce the recovery time in the restoration of the survivable spanning connection. The survivability is nearly optimal.

The rest of this paper is organized as follows. In Sect. 2, we review the previous work. In Sect. 3, we model an optimization problem. In Sect. 4, the algorithm is proposed for solving the SSCR problem. Section 5 evaluates the performance of the proposed algorithm by the simulation of two network topologies. Section 6 concludes this paper.

2 Previous Algorithm

In [4], the Constrained Bandwidth Max-Survivability (CBMS) algorithm is proposed to contribute an optimal set of two spanning trees for the maximization of the survivability level while ensuring the bandwidth.

In CBMS, a transformed network with each link accommodates the bandwidth requirement B_0 is constructed. Then, the CBMS constitutes a set of k -duplicated links to represent a single link e in the transformed network. Accordingly, a matroid is formed by the links of the transformed network [11]. The k link-disjoint spanning trees with minimum cost are generated in the transformed network by applying a matroid greedy algorithm to minimum-cost link disjoint spanning tree [7, 8]. As shown in Fig. 1(a), the network is represented by an undirected graph $G = (V, E)$. Each link is associated with two parameters (p_e, b_e) where p_e is the failure probability of the link and b_e is the bandwidth of the link. $\tilde{G}(V, \tilde{E})$ is the auxiliary network of G , where $\tilde{E} = \{e|e \in E, b_e \geq B_0\}$. Figure 1(a) shows that $\tilde{G}(V, \tilde{E})$ is generated by removing the links with the bandwidth which is less than B_0 . Finally, the result of CBMS shows that the maximum level of survivability can be well-approximated by establishing just two not-necessarily-disjoint spanning trees. In Fig. 1(b), the SSC(T_1, T_2) with the maximum survivability is generated by CBMS. A spanning tree $T_i(V, E_i)$ is the i th spanning tree of G , i.e., $V_i = V$ and $E_i \subseteq E$, for $\forall i \in \{1, 2, \dots, k\}$. Note that the spanning trees in SSC are not necessarily disjoint. Clearly, a pair of spanning trees T_1 and T_2 have two shared links, i.e., $\{(1, 2), (1, 4)\}$.

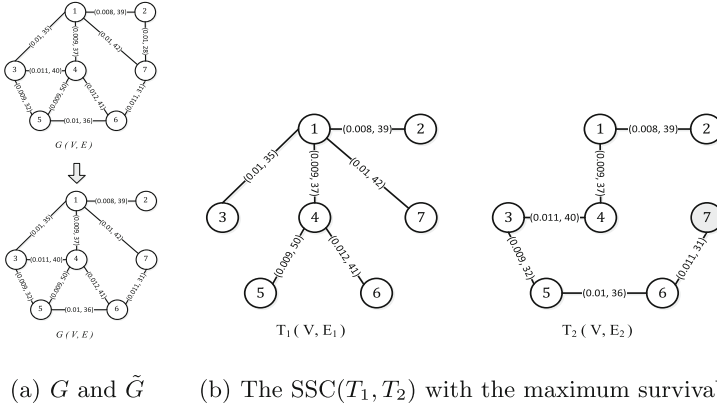


Fig. 1. The network $G(V, E)$, the auxiliary network $\tilde{G}(V, \tilde{E})$ and the $SSC(T_1, T_2)$ with the maximum survivability

The optimal survivable spanning connection for certain sized problems is generated by CBMS. The time complexity is $O(|E| \log(|E|) + |V|^2)$. The survivable spanning connection with T_1 and T_2 is denoted as $SSC(T_1, T_2)$ in this paper.

As reported in [4], the maximum level of survivability can be well-approximated by establishing just two spanning trees in the network with the single link failure model. We follow the case in this paper and the assumption that at most one link failure in the network at a time. The $SSC(T_1, T_2)$ is pre-constructed by CBMS algorithm under QoS requirements. The SSC is failed when the shared link fails between the T_1 and T_2 . In this paper, the fast and efficient restoration algorithm for the SSC is proposed.

3 Definitions and Problem Formulation

Given the network $G = (V, E)$ and the bandwidth requirement B_0 . Each link $e \in E$ is assigned with a failure probability $p_e \in (0, 1]$ and a bandwidth b_e . The p_e s are independent, which are estimated from the failure data of the network. Given the $SSC(T_1, T_2)$ of \tilde{G} , the SSC of \tilde{G} is a tuple of two spanning trees (T_1, T_2) of \tilde{G} . The link $e \in E$ is called a shared link if $e \in T_1 \cap T_2$.

In order to facilitate the understanding of the algorithm, the following describes some definitions.

Definition 1. Given the $SSC(T_1, T_2)$, the bandwidth of the spanning tree $B(T_i)$ is defined as the bandwidth of bottleneck link in T_i , i.e., $B(T_i) = \min_{e \in T_i}(b_e)$. The bandwidth of $SSC(T_1, T_2)$, denoted as $B(T_1, T_2)$, is defined as the bandwidth of the bottleneck spanning tree, i.e.,

$$B(T_1, T_2) = \min\{B(T_1), B(T_2)\} = \min_{e \in T_1 \cup T_2}(b_e). \quad (1)$$

Definition 2. Given the $SSC(T_1, T_2)$, the survivability level $S(T_1, T_2)$ is the feasible probability of all shared links between T_1 and T_2 , i.e.,

$$S(T_1, T_2) = \prod_{e \in T_1 \cap T_2} (1 - p_e). \quad (2)$$

In order to ensure the excellent performance of the network for data transmission, the SSC must have the best survivability at all times. For the $SSC(T_1, T_2)$, $S(T_1, T_2)$ is the maximum survivability level currently in the network. The connection must be promptly restored to a new SSC with the maximum survivability when a shared link fails.

Definition 3. Given the $SSC(T_1, T_2)$, the faulty link $e_f(u, v)$ indicates the link with a permanent fault between the nodes u and v .

Definition 4. Given the $SSC(T_1, T_2)$, the connections $SSC(T'_1, T'_2)$ are called as the (T_1, T_2) —based SSC, where T'_1 and T'_2 are restored from T_1 and T_2 , respectively.

Given the $SSC(T_1, T_2)$ with p_e s, the bandwidth constraint $B_0 > 0$ and $B(T_1, T_2) \geq B_0$, how to restore the SSC by adjusting the invalid spanning trees in the connection when a shared link fails, and generate a (T_1, T_2) —based SSC with close-to-optimal survivability, under the given bandwidth constraint?

The problem can be formalized as follows.

Definition 5. The SSC Restoration (SSCR) Problem: Given the $SSC(T_1, T_2)$ and the faulty shared link $e_f(u, v)$, find a (T_1, T_2) —based SSC such that

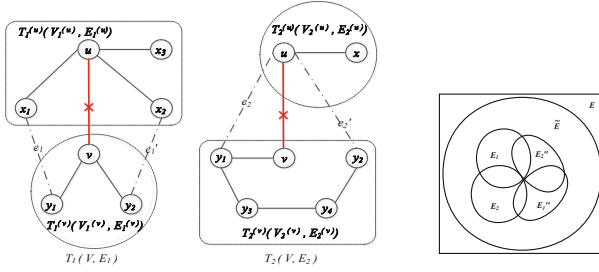
$$\begin{aligned} & \text{Max } S((T_1, T_2)\text{—SSC}) \\ & \text{s.t. } B(S((T_1, T_2)\text{—based SSC})) \geq B_0, \end{aligned}$$

where $S((T_1, T_2)$ —based SSC) is the survivability of the (T_1, T_2) —based SSC, and $B(S((T_1, T_2)$ —based SSC) is the bandwidth of the (T_1, T_2) —based SSC.

4 The SLMS Algorithm

The proposed algorithm to solve the SSCR problem, Shared Links Max-Survivability (SLMS), is proposed to generate a (T_1, T_2) —based SSC by restoring T_1 and T_2 . As shown in Fig. 2(a), T_1 is divided into two subtrees $T_1^{(u)}(V_1^u, E_1^u)$ and $T_1^{(v)}(V_1^v, E_1^v)$, where $u \in V_1^u$ and $v \in V_1^v$. Similarly, T_2 is divided into two subtrees $T_2^{(u)}(V_2^u, E_2^u)$ and $T_2^{(v)}(V_2^v, E_2^v)$, where $u \in V_2^u$ and $v \in V_2^v$.

For the spanning tree T_1 , the set of these $e(x_{i1}, y_{j1})$ s is selected to replace $e_f(u, v)$, which connect $T_1^{(u)}$ with $T_1^{(v)}$, denoted as E_1'' , i.e., $E_1'' = \{e(x_{i1}, y_{j1}) | e(x_{i1}, y_{j1}) \in \tilde{E}, x_{i1} \in V_1^u \text{ and } y_{j1} \in V_1^v, x_{i1} \neq u \text{ or } y_{j1} \neq v, i1 \in N^*, j1 \in N^*\}$. Meanwhile, for T_2 , $E_2'' = \{e(x_{i2}, y_{j2}) | e(x_{i2}, y_{j2}) \in \tilde{E}, x_{i2} \in V_2^u \text{ and } y_{j2} \in V_2^v, x_{i2} \neq$

(a) The subtrees of T_1 and T_2 .

(b) The relationship of the link sets.

Fig. 2. An example of subtrees and corresponding relationship of the link sets.

u or $y_{j_2} \neq v, i_2 \in N^*, j_2 \in N^*\}$. The link in E_1'' and E_2'' is replace the faulty link. Then the spanning trees are connected. Figure 2(b) shows the relationship between the sets $E, \tilde{E}, E_1, E_2, E_1'',$ and E_2'' . There are two cases according to whether the intersection of E_1'' and E_2'' is empty, as follows:

- Case A: $E_1'' \cap E_2'' = \emptyset$. If the intersection of E_1'' and E_2'' is empty, the $e(x_{i_1}, y_{j_1})$ with the minimum failure probability from E_1'' is selected and added into T_1 . The new spanning tree is denoted as $T_1'(V, E_1')$, where $E_1' = \{E_1 - e_f(u, v) + e(x_{i_1}, y_{j_1}) \mid e(x_{i_1}, y_{j_1}) \in E_1'', \min p_{e(x_{i_1}, y_{j_1})}\}$. At the same time, the $e(x_{i_2}, y_{j_2})$ with the minimum failure probability from E_2'' is selected and then added into T_2 . The new spanning tree is denoted as $T_2'(V, E_2')$, where $E_2' = \{E_2 - e_f(u, v) + e(x_{i_2}, y_{j_2}) \mid e(x_{i_2}, y_{j_2}) \in E_2'', \min p_{e(x_{i_2}, y_{j_2})}\}$. As a result, the $\text{SSC}(T_1, T_2)$ is restored to $\text{SSC}(T_1', T_2')$. The number of the shared links in $\text{SSC}(T_1', T_2')$ is more than the original $\text{SSC}(T_1, T_2)$, because the selected links are not in $E_1'' \cap E_2''$. Then the survivability of $\text{SSC}(T_1', T_2')$ defined as

$$S(T_1', T_2') = \frac{(1 - p_{e(x_{i_1}, y_{j_1})})(1 - p_{e(x_{i_2}, y_{j_2})})}{(1 - p_{e_f(u, v)})} S(T_1, T_2). \quad (3)$$

- Case B: $E_1'' \cap E_2'' \neq \emptyset$. If the intersection of E_1'' and E_2'' is not empty, the links in $E_1'' \cap E_2''$ are shared by T_1 and T_2 . The link $e(x_i, y_j)$ with the minimum failure probability from $E_1'' \cap E_2''$ is selected and added into both T_1 and T_2 , and the new $\text{SSC}(T_1', T_2')$ is formed. The spanning trees of $\text{SSC}(T_1', T_2')$ are denoted as $T_1'(V, E_1')$ and $T_2'(V, E_2')$, respectively, where $E_1' = \{E_1 - e_f(u, v) + e(x_i, y_j) \mid e(x_i, y_j) \in E_1'' \cap E_2'', \min p_{e(x_i, y_j)}\}$ and $E_2' = \{E_2 - e_f(u, v) + e(x_i, y_j) \mid e(x_i, y_j) \in E_1'' \cap E_2'', \min p_{e(x_i, y_j)}\}$. The number of the shared links is not change and the survivability of $\text{SSC}(T_1', T_2')$ defined as

$$S(T_1', T_2') = \frac{(1 - p_{e(x_i, y_j)})}{(1 - p_{e_f(u, v)})} S(T_1, T_2). \quad (4)$$

The formula description of the algorithm is omitted due to the limit of paper length.

Theorem 1: Given the network $\tilde{G}(V, \tilde{E})$ with the faulty shared link $e_f(u, v)$ and the SSC (T_1, T_2) with maximum survivability, the (T_1, T_2) —based SSC (T'_1, T'_2) with maximum survivability is produced by SLMS, if $e_f(u, v)$ is not the bridge of network G .

Proof. The (T_1, T_2) —based SSC (T'_1, T'_2) exists, because $e_f(u, v)$ is not the bridge of network G . The faulty link $e_f(u, v)$ splits the spanning tree T_1 into the subtrees $T_1^{(u)}$ and $T_1^{(v)}$ and splits the spanning tree T_2 into the subtrees $T_2^{(u)}$ and $T_2^{(v)}$, respectively. According to the definition of E''_1 and E''_2 , there are not empty. Thus, $e_f(u, v)$ is replaced by the link, which is selected from E''_1 and E''_2 by SLMS. Noting that the link with minimum failure probability is selected in SLMS, thus the (T_1, T_2) —based SSC (T'_1, T'_2) with maximum survivability is found by SLMS.

Theorem 2: The time complexity of SLMS is $O(|E|)$.

Proof. In SLMS, the split of tree T_1 and T_2 runs in constant time complexity. Noting that $|E''_1| < |\tilde{E}| \leq |E|$ and $|E''_2| < |\tilde{E}| \leq |E|$, it is confirmed that E''_1 and E''_2 can be set up in $O(|E|)$. The selection of the smallest element in the set of size $|E|$ can be performed in $O(|E|)$. The insertion of the selected element runs in constant time complexity. Thus, the time complexity of SLMS is $O(|E|)$.

5 Simulation Results

In this section, the simulation results demonstrate the advantages of the algorithm SLMS compared to the algorithm CBMS for the restoration of SSC. The simulations examine the impacts of the link failure probability distribution and network topology. The algorithms are evaluated with two parameters, i.e., recovery time and survivability. The results are the average of the 100 random simulations with various network sizes.

5.1 Simulation Setup

In all simulations, the random networks are generated with different sizes, i.e., $|V| = 10, 20, 30, 50, 70, 100$. The bandwidth of the link in the range of $[5, 150]$ MB/s with a uniform distribution and the p_e obeys two different distributions, i.e., normal distribution and exponential distribution.

Network Topology: Two types of network topologies, namely Power-Law topology [12] and Waxman topology [13]. The network topologies are generated randomly.

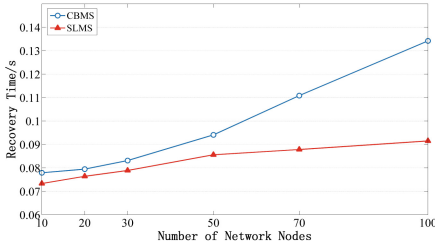
For the Power-Law topology, the outdegree of each node is randomly assigned by the power-law distribution $\beta \cdot x^{-\alpha}$, where x is a random out of the number of network nodes, $\alpha = 0.61$ and $\beta = 100$. Specifically, the pair of the nodes m and n is picked, and the node n keeps the outdegree credits. Then the link between the nodes m and n is added, and the outdegree credit of the node

m is decreased [12]. The generation of the Waxman topology is specified. The source and the destination are located at the diagonally opposite corners of a square of unit dimension. Then, $|V| - 2$ nodes randomly spread over the square. For each pair of nodes m and n , a link (m, n) with the following probability: $p(m, n) = \alpha \cdot \exp\left(-\frac{\delta(m, n)}{\beta \cdot \sqrt{2}}\right)$ where $\alpha = 1$, $\beta = 0.058$ and $\delta(m, n)$ is the distance between the nodes [13].

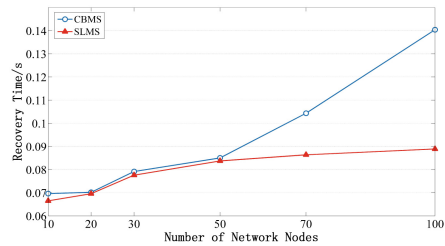
Distributions for p_e : The p_e obeys two different distribution, one is the normal distribution with the mean of 0.01 and the standard deviation of 0.003 and the other is the exponential distribution with the mean of 0.01. The bandwidth constraint is 30.

When the algorithm is executed to the generated networks, $\tilde{E} = \{e | e \in E, b_e \geq 30\}$ is obtained. As mentioned, the faulty link is selected randomly, the algorithm SLMS is used for the SSCR problem with the faulty shared link.

5.2 Recovery Time

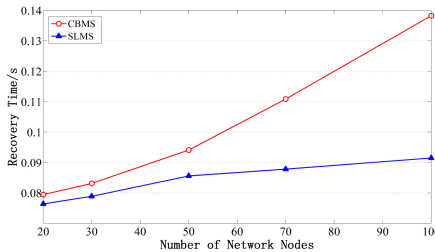


(c) p_e obeys normal distribution

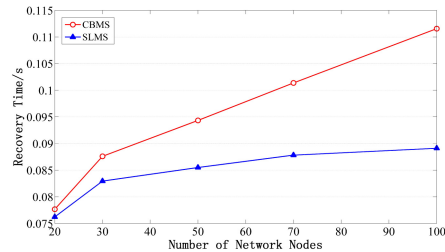


(d) p_e obeys exponential distribution

Fig. 3. Comparison of recovery time art for the faulty shared link, under different distributions of p_e .



(a) In Power-Law Topology



(b) In Waxman Topology

Fig. 4. Comparison of recovery time art for the faulty shared link on different network topologies.

The simulations are implemented for different cases of the SSCR problem and the results are illustrated in Fig. 3 and Fig. 4. The average recovery time of an algorithm \mathcal{A} is denoted as $art(\mathcal{A})$. The average survivability of the restored SSC by an algorithm \mathcal{A} is denoted as $ars(\mathcal{A})$.

In Power-law topology, Fig. 3 shows $art(\text{CBMS})$ and $art(\text{SLMS})$ with the faulty shared link under the different distribution of p_e , the growth of $art(\text{USLMS})$ is similar to that of $art(\text{CBMS})$. Compared with the $art(\text{SLMS})$, the $art(\text{CBMS})$ is increased significantly with increasing number of nodes. When p_e obeys exponential distribution with $|V| = 100$, the $art(\text{SLMS})$ and $art(\text{CBMS})$ are 0.0889 s and 0.1404 s, respectively. Figure 4 illustrates $art(\text{CBMS})$ and $art(\text{SLMS})$ for the faulty shared link in different topologies. In Power-Law topology with $|V| = 100$, the $art(\text{SLMS})$ and $art(\text{CBMS})$ are 0.0915 s and 0.1382 s, respectively. In Waxman topology with $|V| = 100$, the $art(\text{SLMS})$ and $art(\text{CBMS})$ are 0.089 s and 0.112 s, respectively.

It is observed that the growth of $art(\text{SLMS})$ is slower than $art(\text{CBMS})$ in different topologies. Compared to the algorithm CBMS, the improvement of an algorithm \mathcal{A} in recovery time is calculated as follows.

$$imp = \frac{art(\mathcal{A}) - art(\text{CBMS})}{art(\text{CBMS})} \times 100\%. \quad (5)$$

Thus, the maximum imp is 36.69% for the faulty shared link.

5.3 Survivability

Table 1. Comparison in survivability ars for Power-Law topology

The faulty link	Parameters	Results in different size of the network					
		10	20	30	50	70	100
Shared faulty links	$avs(\text{CBMS})$	0.984077	0.973546	0.965774	0.957787	0.925089	0.886275
	$avs(\text{SLMS})$	0.983296	0.970279	0.965774	0.954038	0.919654	0.886275
	$loss\ rate$	0.0793%	0.3356%	0	0.3914%	0.5857%	0

Table 2. Comparison in survivability ars for Waxman topology

The faulty link	Parameters	Results in different size of the network				
		20	30	50	70	100
Shared faulty links	$avs(\text{CBMS})$	0.990846	0.981406	0.985073	0.989825	0.983394
	$avs(\text{SLMS})$	0.990846	0.981406	0.980956	0.984739	0.978147
	$loss\ rate$	0	0	0.418%	0.5138%	0.5336%

The survivability is one of the most important metrics to evaluate the performance of the algorithm for generating the SSC. The survivability of the different algorithms in Power-Law and Waxman are shown in Table 1 and Table 2, respectively. The loss rate of survivability in the algorithm \mathcal{A} over the algorithm CBMS is calculated as follows.

$$\text{loss rate} = \frac{\text{ars}(\mathcal{A}) - \text{ars}(\text{CBMS})}{\text{ars}(\text{CBMS})} \times 100\%. \quad (6)$$

When the SSC is generated by CBMS, the survivability is maximum currently. It is observed that the survivability of the SSC, which is generated by the proposed algorithm, is close to the survivability of the SSC generated by CBMS. For the the faulty shared link, the survivability of the SSC generated by SLMS is only slightly lower than CBMS. The maximum of the *loss rates* are 0.5857% and 0.5336% in Power-Law and Waxman, respectively. However, the *imps* are 20.76% and 20.13% at a time, respectively.

6 Conclusion

The survivable spanning connection is a novel protection scheme for coping with network failures, which is established by two not-necessarily-disjoint spanning trees. In this paper, we have modeled an optimization problem on restoring the survivable spanning connection with the faulty links. Then, we have presented the algorithm to restore the connection rapidly, where the faulty link is shared by the two spanning trees. Simulation results show that the proposed algorithm can reduce the recovery time by up to 36% for the faulty shared link, while keeping the close-to-optimal survivability of the new survivable spanning connection. The distributed algorithm for restoring the survivable spanning connection will be future work.

References

1. Christopher, R.C.: 100-Gb/s and beyond transceiver technologies. *Opt. Fiber Technol.* **17**(5), 472–479 (2011)
2. Zhou, Z., Lin, T.: Survivable Probability of SDN-enabled Cloud Networking with Random Physical Link Failure. *Computer Science: Networking and Internet Architecture* (2017)
3. Wang, R., Gao, S., Yang, W., Jiang, Z.: Energy aware routing with link disjoint backup paths. *Comput. Netw. Int. J. Comput. Telecommun. Netw.* **115**, 42–53 (2017)
4. Yallouz, J., Rottenstreich, O., Orda, A.: Tunable survivable spanning trees. *IEEE/ACM Trans. Netw.* **24**(3), 1853–1866 (2016)
5. Moussaoui, A., Boukeream, A.: A survey of routing protocols based on link stability in mobile ad hoc networks. *J. Netw. Comput. Appl.* **47**, 1–10 (2015)
6. ITU-T G.8032: Ethernet Ring Protection Switching (2015)
7. Roskind, J., Tarjan, R.E.: A note on finding minimum-cost edge-disjoint spanning trees. *Math. Oper. Res.* **10**(4), 701–708 (1985)

8. Clausen, J., Hansen, L.A.: Finding k edge-disjoint spanning trees of minimum total weight in a network: an application of matroid theory. In: Rayward-Smit, V.J. (ed.) *Combinatorial Optimization II. Mathematical Programming Studies*, vol. 13, pp. 88–101. Springer, Heidelberg (1980). <https://doi.org/10.1007/BFb0120910>
9. AlBdaiwi, B., Hussain, Z., Cerny, A., Aldred, R.: Edge-disjoint node-independent spanning trees in dense gaussian networks. *J. Supercomput.* **72**(12), 4718–4736 (2016). <https://doi.org/10.1007/s11227-016-1768-x>
10. Yallouz, J., Orda, A.: Tunable QoS-aware network survivability. *IEEE/ACM Trans. Netw.* **25**(1), 139–149 (2017)
11. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: *Introduction to Algorithms*, 3rd edn. MIT Press, Cambridge (2009)
12. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the Internet topology. *Comput. Netw.* **29**(4), 251–262 (1999)
13. Waxman, B.: Routing of multipoint connections. *IEEE J. Sel. Areas Commun.* **6**(9), 1617–1622 (1988)



On the Decycling Problem in a Torus

Antoine Bossard^(✉) 

Graduate School of Science, Kanagawa University, Tsuchiya 2946, Hiratsuka,
Kanagawa 259-1293, Japan
abossard@kanagawa-u.ac.jp

Abstract. The torus topology has proven very popular as the interconnection network of modern supercomputers. In fact, the world number one as of November 2020, the Supercomputer Fugaku, relies on this network topology. Considering the large number of computing nodes (millions), efficient parallel processing is key to maximise the performance. It is well known that parallel processing is hampered by cycles, deadlocks and starvations being two notorious issues directly linked to the presence of cycles. Hence, network decycling is a critical problem and it has been extensively discussed in the literature. In this paper, we propose a decycling algorithm for the torus topology and compare it with established results.

Keywords: Interconnection network · Algorithm · Parallel processing · Cycle · Feedback vertex set

1 Introduction

Modern supercomputers include hundreds of thousands of computing nodes, with the most recent machines having several millions (e.g. 10,649,600 nodes for the Sunway TaihuLight as of November 2020's TOP500 list [16]). Node interconnection is thus a critical issue to maximise the parallel processing performance. Thanks to its advantageous topological properties (e.g. regularity), the torus topology has proven very popular as the interconnection network of recent supercomputers: the world number one supercomputer as per the November 2020 TOP500 ranking, the Supercomputer Fugaku built by Fujitsu and RIKEN, is relying on the torus topology to connect its nodes (Tofu interconnect D [2]). The IBM Blue Gene/L and Blue Gene/P, Cray Titan (Gemini interconnect [3]) and Fujitsu SORA-MA (Tofu interconnect 2 [1]) are additional examples of torus-based supercomputers.

It is common knowledge that the presence of cycles harms parallel processing as they are at the source of the notorious resource allocation issues that are deadlocks, livelocks and starvations [9]. Notably considering this application, the decycling problem, also called the minimum feedback vertex set problem, has thus been largely discussed in the literature. Karp has shown that finding a decycling set of minimum size (i.e. an optimal decycling set) in any graph is

NP-complete [11]. For example, Fomin et al. have described an algorithm that solves this problem in any graph in $O(1.7548^n)$ time [10]. In addition, polynomial solutions have been proposed for particular classes of graphs such as 3-regular graphs [12], convex bipartite graphs [14], permutation graphs [13] and hypercube-based networks [7, 8]. Amongst others, the size of an optimal decycling set (i.e. the decycling number) in the case of cubes and grids has been discussed in [4, 5], and for hypercubes in [15]. In this paper, we propose a polynomial time decycling algorithm for a torus network. It should be noted that while the case of a grid as cited previously seems close to the torus case which is investigated hereinafter, the wrap-around edges of the torus invalidate the grid decycling approach (see the next section for details).

The rest of this paper is organised as follows. Notations and definitions are recalled in Sect. 2. The 1-dimensional torus trivial case is briefly addressed in Sect. 3. The 2-dimensional case is discussed in Sect. 4. Evaluation is conducted in Sect. 5 and concluding remarks are given in Sect. 6.

2 Preliminaries

In this section, notations, definitions and previously established results are recalled. The set of the vertices of a graph G is denoted by $V(G)$, and the set of its edges by $E(G)$. A path in a graph G is a subgraph of G that is an alternating sequence of distinct vertices and edges. Such a vertex–edge sequence but whose two terminal vertices are the same vertex is called a cycle. The length of a path or cycle is its number of edges. A graph that contains no cycle is said to be acyclic and is isomorphic to a tree.

Definition 1. *An n -dimensional k -ary torus, denoted by $T(n, k)$, with $n \geq 1$ and $k \geq 1$, consists in the k^n vertices induced by the set $\{0, 1, \dots, k-1\}^n$. Two vertices $u = (u_0, u_1, \dots, u_{n-1})$ and $v = (v_0, v_1, \dots, v_{n-1})$ of a $T(n, k)$ are adjacent if and only if there exists j ($0 \leq j < n$) such that $\forall i$ ($0 \leq i < n, i \neq j$) $u_i = v_i$ and $u_j = v_j \pm 1 \pmod{k}$.*

A torus $T(n, k)$ is thus a regular graph of degree $2n$, of diameter $n\lfloor k/2 \rfloor$ and has nk^n edges. We next state an important torus property.

Property 1. For a dimension δ ($0 \leq \delta < n$), a $T(n, k)$ consists in k sub-tori $T^{i,\delta}(n-1, k)$ ($0 \leq i < k$). Each sub-torus $T^{i,\delta}(n-1, k)$ is induced by the k^{n-1} vertices $(u_0, u_1, \dots, u_{\delta-1}, i, u_{\delta+1}, \dots, u_{n-1})$ of $T(n, k)$ with u_j ($0 \leq j < n, j \neq \delta$) the vertex coordinate for the dimension j and i the vertex coordinate for the dimension δ .

A sample torus $T(2, 3)$ is given in Fig. 1a and its recursive structure is detailed in Fig. 1b.

A lower bound on the size of a decycling set in any graph has been established by Beineke and Vandell in [6]. The corresponding result is given in Theorem 1 below.

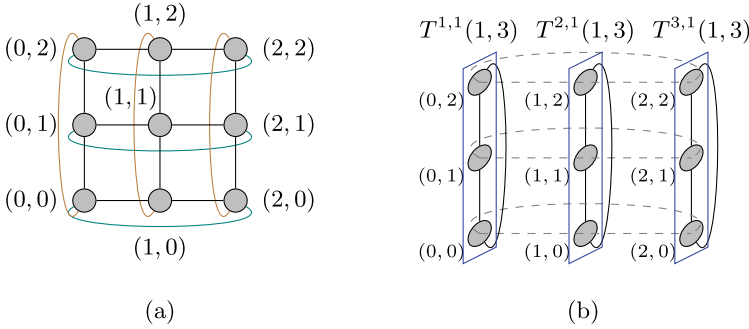


Fig. 1. (a) A torus $T(2,3)$. (b) A torus has a recursive structure: a $T(2,3)$ consists in three sub-tori $T(1,3)$.

Theorem 1. [6] *For a graph $G = (V, E)$ of maximum degree Δ , any decycling set S of G satisfies the following relation:*

$$|S| \geq \left\lceil \frac{|E| - |V| + 1}{\Delta - 1} \right\rceil$$

3 The Case of a $T(1, k)$

This simple case is a gentle concrete introduction to the torus decycling problem. Because, by Definition 1, a $T(1, k)$ is isomorphic to a ring, it is trivial to find a decycling set S : for any vertex u in $T(1, k)$, define $S = \{u\}$.

This decycling set S is clearly optimal, and this can also be verified with Theorem 1: by Definition 1, the number of vertices of a $T(1, k)$ is equal to k , the number of edges is also equal to k , the maximum degree is 2 and therefore the lower bound on the size of a decycling set of Theorem 1 is equal to 1, which is the cardinality of the obtained decycling set. Hence, S is an optimal decycling set. An illustration is given in Fig. 2.

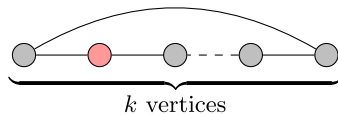


Fig. 2. The trivial case $T(1, k)$: an optimal decycling set is of cardinality one, for instance the red vertex. (Colour figure online)

4 The Case of a $T(2, k)$

We describe in this section the details of our approach to decycle a 2-dimensional k -ary torus $T(2, k)$. We give below a constructive proof in the form of a decycling

algorithm whose input is an arity k ($k \geq 1$) and which outputs a decycling set S . The main idea of the algorithm is to first consider a “suboptimal” decycling set and to subsequently carefully restore vertices so that the obtained graph remains acyclic.

4.1 Special Cases

The case $k = 1$ is trivial: $T(2, 1)$ consists of one unique vertex and is thus acyclic; $S = \emptyset$ is thus a decycling set for this trivial graph.

When $k = 2$, because it is conventionally assumed that the graph has no double edges, the same discussion as that of Sect. 3 can be held: the singleton set $S = \{u\}$ with u any vertex of $T(2, 2)$ is a decycling set. (If for some peculiar reason double edges do exist, a set S is a decycling set if and only if it consists of two diagonally opposed vertices, for instance $S = \{(0, 0), (1, 1)\}$.)

When $k = 3$, the set $S = \{(0, 2), (1, 1), (2, 0), (2, 2)\}$ is a decycling set since the induced subgraph \tilde{T} of $T(2, 3)$ defined by the vertices $V(\tilde{T}) = V(T(2, 3)) \setminus S$ has two components: a vertex of degree 0 and a path (of length 3).

The decycling sets described in this section are optimal by Theorem 1. The latter three cases are illustrated in Fig. 3.

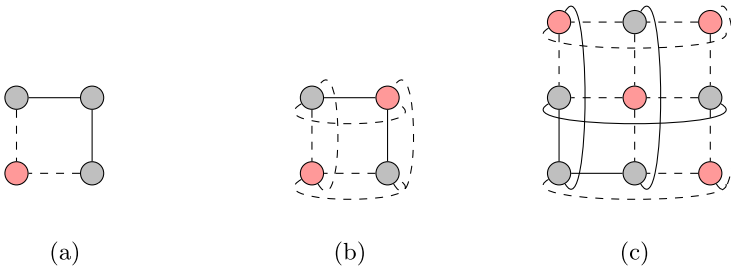


Fig. 3. Optimal decycling sets (red vertices) when (a) $k = 2$, (b) $k = 2$ with double edges and (c) $k = 3$. (Colour figure online)

4.2 General Case

We can assume that $k \geq 4$ as the other cases have been treated in Sect. 4.1. The proposed decycling algorithm consists of the following three main steps. Step 1 is illustrated in Fig. 4, Step 2 in Fig. 5 and Step 3 in Fig. 6, each time illustrating side-by-side the cases k even and k odd.

Step 1. Define a first (suboptimal) decycling set $\tilde{S} \subset V(T(2, k))$ as follows:

$$\tilde{S} = \{(i, j) \mid 0 \leq i, j \leq k - 1, i + j \equiv 0 \pmod{2}\}$$

In other words, \tilde{S} is induced by the vertices of a $T(2, k)$ that are taken in one particular “quincunx” manner. The induced subgraph \tilde{T} of $T(2, k)$ defined by the vertices $V(\tilde{T}) = V(T(2, k)) \setminus \tilde{S}$ is indeed acyclic since $E(\tilde{T}) = \emptyset$ is satisfied.

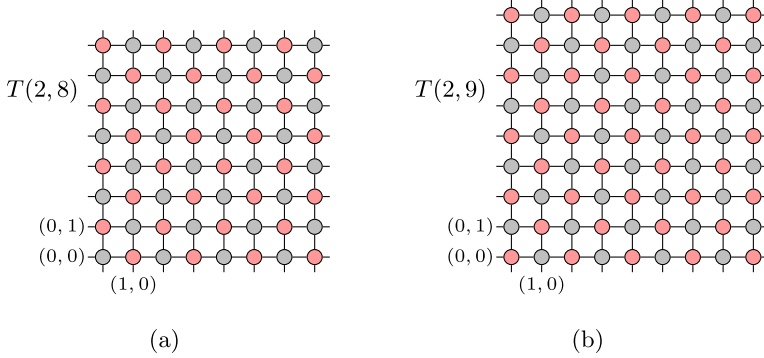


Fig. 4. Illustration of the algorithm’s Step 1: (a) in a $T(2, 8)$; (b) in a $T(2, 9)$. The red vertices are in \tilde{S} . (Colour figure online)

Step 2. Next, we select vertices to be restored, that is, vertices that are included in \tilde{S} but that will be excluded from the final decycling set to further reduce its size. Three sub-steps are distinguished.

Step 2.1. For the sake of clarity, we assume without loss of generality that the vertices of a $T(2, k)$ are arranged according to a $k \times k$ grid, say of X and Y axis. The vertices of the set \tilde{S} induce vertex diagonals (i.e. a set of vertices) on this grid, say the set D_1 that includes the vertex diagonals of slope -1 and the set D_2 that includes the vertex diagonals of slope 1 . We consider the $k - 1 + (k \bmod 2)$ vertex diagonals d_i of the set D_1 ($0 \leq i < |D_1|$) where d_i includes the vertex $(i, i + 1 - (k \bmod 2))$. For example, when k is even the vertex diagonal d_0 consists of the vertices $\{(0, 1), (1, 0)\}$ and when k is odd of the vertex $\{(0, 0)\}$.

The number of vertices $|d_i|$ in a diagonal d_i ($0 \leq i < |D_1|$) is given by the following formula:

$$|d_i| = \begin{cases} 2i + 2 & \text{if } k \text{ even, } 0 \leq i < \frac{k}{2} \\ 2i + 1 & \text{if } k \text{ odd, } 0 \leq i \leq \frac{k-1}{2} \\ 2(k - i - 1) & \text{if } k \text{ even, } \frac{k}{2} \leq i < k - 1 \\ 2(k - i) - 1 & \text{if } k \text{ odd, } \frac{k-1}{2} < i < k \end{cases}$$

Step 2.2. Define the vertex diagonal set $D \subset D_1$ as $\{d_i \mid 0 \leq i < |D_1|, i \text{ odd}\}$.

In other words, we alternately select the diagonals of D_1 . The equality $|D| = \lfloor (k - 1)/2 \rfloor$ holds.

Step 2.3. Given a vertex diagonal $d \in D$, let \vec{d} be the ordered set whose vertices are those of d and which are ordered in ascending order of their X coordinates. So, for example, when k is odd the ordered set that corresponds to d_1 is $\vec{d}_1 = ((0, 2), (1, 1), (2, 0))$. Furthermore, given a vertex diagonal $d \in D$, define the vertex set

$$R_d = \left\{ u_i \mid (u_0, u_1, \dots, u_{|d|}) = \vec{d}, 0 \leq i \leq |d|, i \equiv k \pmod{2} \right\}$$

In other words, the vertices of the set R_d consist of the alternating vertices of the diagonal d .

From this discussion, we can define R_1 the first subset of \tilde{S} that will be used to restore vertices as explained. We have

$$R_1 = \bigcup_{d \in D} R_d$$

The induced subgraph \tilde{T} of $T(2, k)$ defined by the vertices $V(\tilde{T}) = V(T(2, k)) \setminus (\tilde{S} \setminus R_1)$ is indeed acyclic since the edges at the restored vertices (i.e. the vertices of R_1) induce vertex-disjoint stars (of four edges) or vertex-disjoint trees each induced by a 4-edge star with one or two additional vertices.

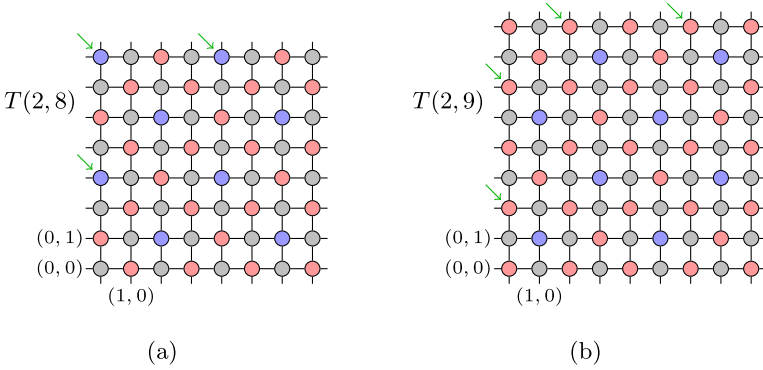


Fig. 5. Illustration of the algorithm's Step 2: (a) in a $T(2, 8)$; (b) in a $T(2, 9)$. The red vertices are in \tilde{S} and the blue ones in R_1 . The selected diagonals are indicated with an arrow. (Colour figure online)

Step 3. Finally, we restore additional vertices. If $k \in \{4, 5, 7\}$, define $R_2 = \emptyset$. If $k = 6$, define $R_2 = \{(0, 1)\}$. Otherwise, define the set R_2 as described in the rest of this step.

Let $\delta = 0$ be the dimension used to reduce $T(2, k)$ into k sub-tori $T^{i, \delta}(1, k)$ ($0 \leq i \leq k-1$) as per Property 1. Define T a subset of the k sub-tori $T^{i, \delta}(1, k)$ ($0 \leq i \leq k-1$) as follows:

$$T = \left\{ T^{4i + (k \bmod 2), \delta}(1, k) \mid 0 \leq i < \left\lceil \frac{k - (k \bmod 2)}{4} \right\rceil \right\}$$

For each sub-torus $T_1 \in T$, one vertex of $T_1 \cap (\tilde{S} \setminus R_1)$ is restored (i.e. added into R_2) as per the following definition:

$$R_2 = \{(4i + (k \bmod 2), 4i + 1 + 2(k \bmod 2)) \mid 0 \leq i < |T|\}$$

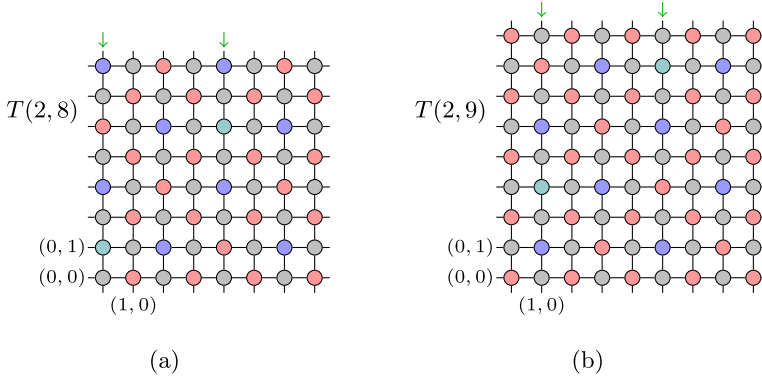


Fig. 6. Illustration of the algorithm’s Step 3: (a) in a $T(2, 8)$; (b) in a $T(2, 9)$. The red vertices are in \tilde{S} , the blue ones in R_1 and the green ones in R_2 . The sub-tori of T are indicated with an arrow. (Colour figure online)

The final decycling set S is given by

$$S = \tilde{S} \setminus (R_1 \cup R_2)$$

The induced subgraph \tilde{T} of $T(2, k)$ defined by the vertices $V(\tilde{T}) = V(T(2, k)) \setminus S$ is indeed acyclic since the edges at the restored vertices (i.e. the vertices of $R_1 \cup R_2$) induce vertex-disjoint 4-edge stars or vertex-disjoint trees induced by 4-edge stars.

5 Evaluation

We establish in this section the complexities of the method proposed in Sect. 4; it is trivial for Sect. 3.

5.1 Cardinalities and Time Complexity

We distinguish the two cases k even and k odd.

k even Regarding Step 1, the set \tilde{S} has $2(k/2 \times k/2) = k^2/2$ vertices. The set \tilde{S} can thus be calculated in $O(k^2)$ time.

Regarding Step 2, the set R_1 has $\lceil k/4 \rceil \lfloor k/4 \rfloor + \lceil k/4 \rceil \lfloor k/4 \rfloor = 2\lceil k/4 \rceil \lfloor k/4 \rfloor$ vertices. In Steps 2.1 and 2.2, $|D| = (k - 2)/2$ vertex diagonals are effectively gathered, which thus requires $O(k)$ time. In Step 2.3, each of these $(k - 2)/2$ vertex diagonals is iterated. The sum of their number of vertices is equal to $(k/2)^2 - ((k/2) \bmod 2)$ and this takes thus $O(k^2)$ time in total.

Regarding Step 3, the set R_2 has zero vertex when $k = 4$, one vertex when $k = 6$ and $\lceil k/4 \rceil$ vertices otherwise. This step takes thus $O(k)$ time.

k odd Regarding Step 1, the set \tilde{S} has $(k+1)/2 \times (k+1)/2 + (k-1)/2 \times (k-1)/2 = (k^2 + 1)/2$ vertices. The set \tilde{S} can thus be calculated in $O(k^2)$ time.

Regarding Step 2, the set R_1 has $\lceil (k-1)/4 \rceil \lceil (k-1)/4 \rceil + \lfloor (k-1)/4 \rfloor \lfloor (k-1)/4 \rfloor = \lceil (k-1)/4 \rceil^2 + \lfloor (k-1)/4 \rfloor^2$ vertices. In Steps 2.1 and 2.2, $|D| = (k-1)/2$ vertex diagonals are effectively gathered, which thus requires $O(k)$ time. In Step 2.3, each of these $(k-1)/2$ vertex diagonals is iterated. The sum of their number of vertices is equal to $(k+1)/2 \times (k-1)/2 + ((k-1)/2 \bmod 2)$ and this takes thus $O(k^2)$ time in total.

Regarding Step 3, the set R_2 has zero vertex when $k \leq 7$ and $\lceil (k-1)/4 \rceil$ vertices otherwise. This step takes thus $O(k)$ time.

5.2 Main Result

The discussion of Sect. 5.1 is summarised in the following theorem.

Theorem 2. *In a 2-dimensional k -ary torus $T(2, k)$ ($k \geq 1$), a decycling set S of 0 vertex when $k = 1$, 1 vertex when $k = 2$, 4 vertices when $k = 3$ and with*

$$|S| = \begin{cases} k^2/2 - 2\lceil k/4 \rceil \lfloor k/4 \rfloor & k = 4 \\ (k^2 + 1)/2 - \lceil (k-1)/4 \rceil^2 - \lfloor (k-1)/4 \rfloor^2 & k = 5, k = 7 \\ k^2/2 - 2\lceil k/4 \rceil \lfloor k/4 \rfloor - 1 & k = 6 \\ k^2/2 - 2\lceil k/4 \rceil \lfloor k/4 \rfloor - \lceil k/4 \rceil & k \geq 8, k \text{ even} \\ (k^2 + 1)/2 - \lceil (k-1)/4 \rceil^2 - \lfloor (k-1)/4 \rfloor^2 - \lceil (k-1)/4 \rceil & k \geq 9, k \text{ odd} \end{cases}$$

in the other cases can be found in $O(k^2)$ time.

Proof. A constructive proof of such a set S has been described in Sect. 4. Furthermore, the cardinality of S is equal to $|\tilde{S}| - |R_1| - |R_2|$ since $R_1 \cap R_2 = \emptyset$ by definition and $|\tilde{S}|, |R_1|, |R_2|$ have been calculated in Sect. 5.1. In the same section, it has been shown that such a set S can be found with a worst-case time complexity of $O(k^2)$.

5.3 Comparison with the Lower Bound

In this section, we investigate how close the size of the decycling set generated is to the lower bound of Theorem 1. The values obtained from Theorems 1 and 2 are represented in Fig. 7. It is recalled that the result of Theorem 1 is a lower bound on the size of a decycling set, and not necessarily the size of an optimal decycling set. So, the difference plotted in Fig. 7 is given for reference, and it shows that the size of the obtained decycling set is promising, possibly optimal in some cases, given that it is rather close, and sometimes equal, to the lower bound of Theorem 1.

It can also be observed that, obviously, the size of the generated decycling set S is never smaller than the lower bound of Theorem 1; the contrary would indicate a hole in the proposed algorithm.

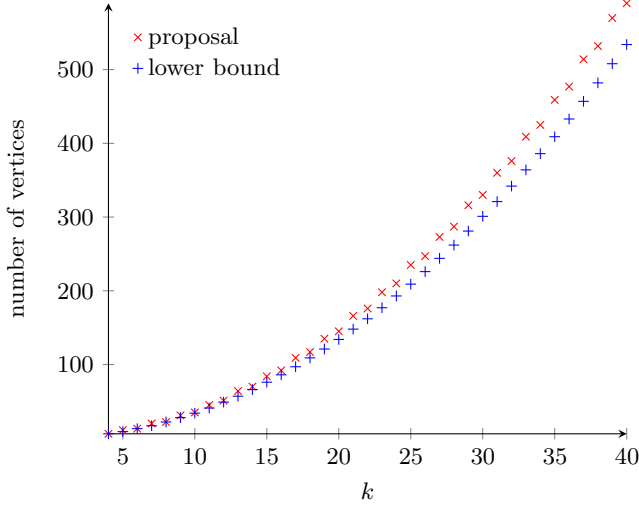


Fig. 7. Investigating how close the size of the decycling set generated is to the lower bound of Theorem 1.

6 Concluding Remarks

The torus topology has been very popular for the interconnection network of massively parallel systems. In fact, the world number one supercomputer as of November 2020, the Supercomputer Fugaku, relies on it. Besides, it is well known that parallel processing is hampered by the presence of cycles in the network of its computing nodes, which is one reason why the decycling problem (NP-complete) has been largely discussed in the literature. In this paper, we have discussed the decycling problem inside a torus $T(n, k)$, and especially in the case $n = 2$. Nevertheless, thanks to the recursive property of the torus topology, this work notably on the decycling of a $T(2, k)$ can be used to render acyclic parts (i.e. 2-dimensional sub-tori) of a torus of higher dimension, which will subsequently facilitate parallel processing as explained. We have given a constructive proof of a decycling set S for a torus $T(2, k)$ where S has $k^2/2 - 2\lceil k/4 \rceil \lfloor k/4 \rfloor - \lceil k/4 \rceil$ vertices when k is even ($k \geq 8$) and $(k^2 + 1)/2 - \lceil (k-1)/4 \rceil^2 - \lfloor (k-1)/4 \rfloor^2 - \lceil (k-1)/4 \rceil$ vertices when k is odd ($k \geq 9$) and can be obtained in $O(k^2)$ time (we treated the cases $n = 1$ and $k \leq 7$ separately), which is promising since rather close to the lower bound on the size of an optimal decycling set.

Regarding future works, further investigating, for instance by means of a computer experiment, how close the size of the calculated decycling set is to that of an optimal decycling set is a first meaningful possibility. Then, it will be very interesting to research, for instance as explained above by relying on the recursive property of the torus topology, how to rely on this result to produce non-trivial decycling sets for tori of higher dimensions.

Acknowledgements. This research was partly supported by a Grant-in-Aid for Scientific Research (C) of the Japan Society for the Promotion of Science under grant no. 19K11887.

References

1. Ajima, Y., et al.: Tofu interconnect 2: system-on-chip integration of high-performance interconnect. In: Kunkel, J.M., Ludwig, T., Meurer, H.W. (eds.) ISC 2014. LNCS, vol. 8488, pp. 498–507. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07518-1_35
2. Ajima, Y., et al.: The Tofu interconnect D. In: Proceedings of the IEEE International Conference on Cluster Computing, pp. 646–654, Belfast, UK, 10–13 September 2018
3. Alverson, R., Roweth, D., Kaplan, L.: The Gemini system interconnect. In: Proceedings of the 18th IEEE Symposium on High Performance Interconnects, pp. 83–87, Mountain View, CA, USA, 18–20 August 2010
4. Bau, S., Beineke, L., Liu, Z., Du, G.M., Vandell, R.: Decycling cubes and grids. *Utilitas Math.* **59**, 129–138 (2001)
5. Bau, S., Beineke, L.W.: The decycling number of graphs. *Australas. J. Comb.* **25**, 285–298 (2002)
6. Beineke, L.W., Vandell, R.C.: Decycling graphs. *J. Graph Theory* **25**(1), 59–77 (1997)
7. Bossard, A.: On the decycling problem in hierarchical hypercubes. *J. Interconnect. Netw.* **14**(2), 1350006.1–1350006.13 (2013)
8. Bossard, A.: The decycling problem in hierarchical cubic networks. *J. Supercomput.* **69**(1), 293–305 (2014). <https://doi.org/10.1007/s11227-014-1152-7>
9. Festa, P., Pardalos, P.M., Resende, M.G.C.: Feedback set problems. In: Du, D.Z., Pardalos, P.M. (eds.) *Handbook of Combinatorial Optimization*, pp. 209–258. Springer, Boston (1999). https://doi.org/10.1007/978-1-4757-3023-4_4
10. Fomin, F.V., Gaspers, S., Pyatkin, A.V.: Finding a minimum feedback vertex set in time $\mathcal{O}(1.7548^n)$. In: Bodlaender, H.L., Langston, M.A. (eds.) *IWPEC 2006*. LNCS, vol. 4169, pp. 184–191. Springer, Heidelberg (2006). https://doi.org/10.1007/11847250_17
11. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds.) *Complexity of Computer Computations*. The IBM Research Symposia Series, pp. 85–103. Springer, Boston (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
12. Li, D., Liu, Y.: A polynomial algorithm for finding the minimum feedback vertex set of a 3-regular simple graph. *Acta Math. Sci.* **19**(4), 375–381 (1999)
13. Liang, Y.D.: On the feedback vertex set problem in permutation graphs. *Inf. Process. Lett.* **52**(3), 123–129 (1994)
14. Liang, Y.D., Chang, M.S.: Minimum feedback vertex sets in cocomparability graphs and convex bipartite graphs. *Acta Informatica* **34**(5), 337–346 (1997)
15. Pike, D.A.: Decycling hypercubes. *Graphs Comb.* **19**(4), 547–550 (2003)
16. TOP500: TOP500 List - November 2020. <https://www.top500.org/lists/top500/list/2020/11/>. Accessed November 2020



Deep Learning Optimization for Many-Core Virtual Platforms

Hengyu Cai¹(✉), Chengming Ning¹, and Qilong Zheng^{1,2}

¹ University of Science and Technology of China, Hefei, China
{chy520,ncmustc}@mail.ustc.edu.cn, qlzheng@ustc.edu.cn

² Anhui Province Key Laboratory of High Performance Computing, Hefei, China

Abstract. The rapid development of deep learning technology has made deep learning models widely used in image processing, speech recognition, and target tracking. However, the model becomes larger and larger, and it is difficult to deploy on a stand-alone device, usually on a distributed computing platform. As a high-performance digital signal processor developed by the 38th Research Institute of China Electronics Technology Group, HXDSP has strong computing power and rich computing resources, and is suitable for computing-intensive applications such as deep learning. Design the many-core virtual platform based on the HXDSP simulator, and provide the parallel communication interface MPIRIO to realize fast communication and task synchronization between the HXDSPs, and provide basic conditions for the deployment of deep learning models. At the same time, the parallel computing capability and pipeline mechanism provided by the virtual platform are used to accelerate the operation of the model. Aiming at the problem that the traditional gradient descent algorithm needs to be manually set, the meta-learning optimization algorithm is used to realize the adaptive fine-tuning of the model on the virtual platform, forming a deep learning optimization framework based on the CPU/HXDSP heterogeneous system.

Keywords: Deep learning · HXDSP · Many-core virtual platform · Meta-learning optimization

1 Introduction

In recent years, the rapid development of deep learning [1] technology has made deep learning models widely used in image processing [2], speech recognition [3], and target tracking [4]. Academia and industry are also studying how to implement and industrialize deep learning technology to improve our life and work efficiency. Mobile devices [5], embedded devices [6], and dedicated accelerators

Supported by the Core Electronic Devices, High-end Generic Chips and Basic Software of National Science and Technology Major Projects of China under Grant No. 2012ZX01034-001-001.

are all preferred platforms. However, due to the increasing size of deep learning models and the limited resources of these computing platforms, the deployment and optimization of deep learning models face challenges.

At present, there are several main solutions to solve the problem that deep learning models are too large and occupy a lot of computing and storage resources. The first solution is to compress the deep learning model [7], thereby reducing the model's occupation of computing and storage resources. The second solution is to deploy the deep learning model on the cloud platform or distributed computing system [8].

Among the many embedded chips, the digital signal processor DSP has strong computing power and low power consumption, and is suitable for computing-intensive applications such as deep learning. The HXDSP [9] series processor is a chip with completely independent intellectual property rights developed by the 38th Research Institute of China Electronics Technology Group Corporation. It has abundant computing and storage resources and has the potential for deep learning application development [10]. In order to realize the deployment and optimization of deep learning models on multiple HXDSP platforms, this paper mainly carries out the following work.

1. Based on the HXDSP chip, the HXDSP many-core virtual platform is designed. The platform uses RapidIO [11] and Ethernet [12] protocol to realize the exchange model, which can carry out fast communication and synchronization.

2. Based on the HXDSP many-core virtual platform, the MPIRIO parallel communication interface is realized by referring to the MPI [13] standard to increase the availability of the virtual platform. MPIRIO provides basic conditions for the deployment of deep learning.

3. The deep learning model can call MPIRIO to generate back-end code that runs on the HXDSP virtual platform. Use the parallel computing capability [14] of HXDSP and the pipeline mechanism [15] provided by the virtual platform for acceleration [16]. A speedup of $10.66\times$ can be obtained on a single-board virtual platform, and a higher speedup on a multi-board virtual platform.

4. Use the meta-learning [17] optimization mechanism to optimize the operation of the model on the virtual platform. The optimizer is deployed on the host side, the optimized model is deployed on the virtual platform, and the shared memory is used for interaction to realize a heterogeneous computing system based on CPU/HXDSP.

2 Background Knowledge

2.1 HXDSP Architecture

The HXDSP1042 chip is a 32-bit DSP processor developed by CETC 38. It integrates two new generation processor cores, eC104+, with a direct communication mechanism between cores. HXDSP has a wealth of peripheral interfaces, supports the RapidIO standard, has an Ethernet interface, and can interconnect

with various types of equipment. The instruction system of the HXDSP chip supports SIMD [18] and VLIW [19] type operations, with multiple instruction issuing slots.

HXDSP has rich peripheral interfaces, high bit width and computing power. A single eC104+ core can reach 30GOPS computing power, the chip can reach 60GOPS, and the peak fixed-point multiplication and accumulation capacity is 32GMACS.

2.2 Deep Learning Model

The deep learning model is mainly composed of a variety of model layers, such as convolutional layer [20], fully connected layer [21] and activation function. Since the convolutional layer occupies most of the calculation in the model, the acceleration is mainly performed on the convolutional layer.

The Fig. 1 shows a convolution operation. The large square on the left in Fig. 1 is the input layer, a 3-channel image with a size of 32×32 . The small square on the left is the convolution kernel filter with a size of 5×5 and a depth of 3. Divide the input layer into multiple regions, use the convolution kernel and each region to perform convolutional operations.

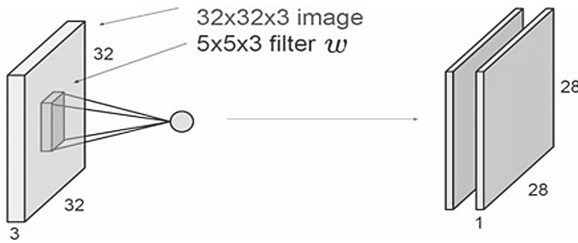


Fig. 1. Schematic diagram of convolution layer

The formula of convolution operation is shown in formula (1)

$$output_{k,i,j} = \sum_{c=0}^2 \sum_{h=0}^4 \sum_{w=0}^4 (filter_{k,h,w,c} \cdot image_{i+h,j+w,c}) \tag{1}$$

2.3 Meta-learning

The meta-learning optimization algorithm is actually a learning mechanism that uses optimizers trained on many old tasks to optimize new tasks.

At present, meta-learning algorithms mainly include metric-based meta-learning and optimization-based meta-learning. The first type of meta-learning is usually a well-learned embedded metric space, which is used to embed unknown task data sets. The second type meta-learning is to train on the old task set to get a better optimizer, which can give the initialization parameters of the new task and optimize it during operation.

3 Realization and Optimization

3.1 The Overall Architecture of the HXDSP Virtual Platform

The single board structure of the HXDSP virtual platform is shown in Fig. 2, which includes two HXDSP chips, a data exchange model and an external memory module. The external memory module can be DDR or other memory modules. The basic structure of the HXDSP many-core virtual platform is a single-board card structure, which forms a larger many-core virtual platform through cascading.

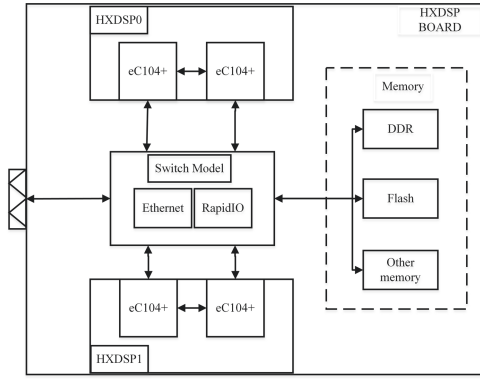


Fig. 2. HXDSP virtual platform architecture

Assuming that the communication bandwidth between cores is dma , and the bandwidth of the data exchange model is $switch$, the $time$ required for the amount of data to be transmitted as $flow$ satisfies formula (2)

$$time = \frac{2 \cdot flow}{2 \cdot dma + switch} \quad (2)$$

3.2 MPIRIO Parallel Communication Interface

MPIRIO parallel communication interface is based on RapidIO communication of HXDSP many-core virtual platform and MPI standard.

The host computer (CPU) distributes the tasks to each chip on the HXDSP board. After each chip completes its own task, it communicates through RapidIO exchange model, and directly exchanges the data to complete the corresponding calculation tasks without sending the data back to the host. Which can reduce the data transmission time and reduce the utilization of host computing resources.

3.3 Model Acceleration Based on Virtual Platform

Algorithm 1 shows the pseudo code for one of the eC104+ cores to perform convolution operations.

Algorithm 1. Convolution based on HXDSP

```

1: for k = 0 to K do
2:   for h = 0 to H do
3:     for w = 0 to W do
4:       clear_XYZTMAC()
5:       for i = 0 to CxRxS do
6:         read_slide_window_data_to_register()
7:         XYZTMAC0+=R2·R0||XYZTMAC1+=R3·R1
8:         ||XYZTMAC2+=R4·R0||XYZTMAC3+=R5·R1
9:       end for
10:      w+=16
11:     write_result_from_register_to_memory()
12:   end for
13: end for
14: end for
15: return flag

```

It is necessary to adjust the memory layout of input data to output multiple convolution results at the same time. Each core of HXDSP chip includes three blocks, and each block includes eight banks. Using dual read bus, multiple elements can be read at the same time, but memory access conflict should be avoided, so interleaved memory layout mode is designed.

Based on HXDSP many-core virtual platform, three-layer pipeline optimization can be realized. They are pipeline between hxdsp chips, pipeline between model layers and pipeline within model layer.

3.4 Meta-learning Based on Virtual Platform

The meta-learning optimizer optimizes the optimizee model with tasks as data sets, and finally obtains the meta-learning optimizer. When a new task is encountered, a better parameter initialization value is given for the new model.

The multi-layer LSTM model is used as the meta-learning optimizer, and the core formula is formula (3) Optimizee will pass the loss and gradient information to the optimizer, and the optimizer will update the optimizee parameters according to the information and its own state. After accumulating some time step loss and gradient information, the optimizer uses Adam optimization algorithm [22] to update its own parameters.

$$\begin{aligned}
\theta^{t+1} &= \theta^t + y^t \\
Y^{t+1} &= \text{forward}(\theta^{t+1} \cdot X^{\text{optimizee}}) \\
F(\theta^{t+1}) &= \frac{1}{2} \cdot (Y^{t+1} - Y^{\text{optimizee}})^2 \\
L(\phi) &= L(\phi) + W^{\text{optimizer}} \cdot F(\theta^{t+1}) \\
X^{\text{optimizer}} &= \nabla_{\theta^{t+1}} = \frac{\partial F(\theta^{t+1})}{\partial \theta^{t+1}} \\
\text{dsigmoid}(y^t) &= (1 - y^t) \cdot y^t \\
\frac{\partial L(\phi)}{\partial \phi} &= \frac{\partial L(\phi)}{\partial H^t} \cdot \frac{\partial H^t}{\partial \phi} \\
\frac{\partial H^t}{\partial \phi^o} &= \tanh(c^t) \cdot \frac{\partial o^t}{\partial \phi^o} \\
\frac{\partial L}{\partial c^t} &= \frac{\partial L(\phi)}{\partial H^t} \cdot o^t \cdot \text{dtanh}(c^t) + \frac{\partial L}{\partial c^{t+1}} \cdot f^{t+1} \\
\frac{\partial L}{\partial \phi^i} &= \frac{\partial L}{\partial c^t} \cdot \tilde{c}^t \cdot \frac{\partial i^t}{\partial \phi^i}
\end{aligned} \tag{3}$$

The deep learning model is deployed on the virtual platform and can be optimized by meta-learning algorithm. The system architecture is shown in Fig. 3. The meta-learning optimizer is deployed on the host side, and the optimized model is deployed on the virtual platform in parallel mode. The HXDSP chip interacts with the host through interruption, and then uses the shared memory for data transmission.

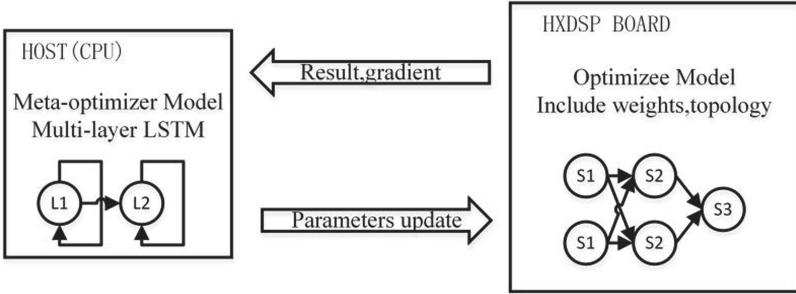


Fig. 3. Meta-learning optimization architecture based on virtual platform

The meta-learning optimization framework algorithm based on CPU/HXDSP heterogeneous system is shown in Algorithm 2.

Algorithm 2. meta-learning algorithm based on CPU/HXDSP

```

1:  $\theta$  is the weights of optimizee model in HXDSP virtual platform
2:  $\phi$  is the weights of optimizer model in CPU
3: initialization of  $\theta$  and  $\phi$ 
4: for epoch = 0 to EPOCHS do
5:   for step = 0 to STEPS do
6:      $\theta_{gradient}, loss_{optimizee} = optimizee(\theta)$ 
7:     transport  $\theta_{gradient}, loss_{optimizee}$  to optimizer
8:      $loss_{optimizer} = loss_{optimizer} + loss_{optimizee}$ 
9:     if step%unroll_length==0 then
10:       $\phi_{gradient} = optimizer(\phi, state, loss_{optimizer})$ 
11:       $\phi = \phi - lr \cdot \phi_{gradient}$ 
12:     end if
13:      $\theta_{update}, state_{update} = optimizer(\phi, state, \theta)$ 
14:      $state = state_{update}$ 
15:     transport  $\theta_{update}$  to optimizee
16:      $\theta = \theta + \theta_{update}$ 
17:   end for
18: end for

```

4 Experimental and Analysis

Our implementation is based on HXDSP1042, and its working frequency is 500 MHz. Software implementation runs on an Intel(R) Core i5-2400 CPU (@3.10 GHz \times 4) with 8GB memory.

4.1 HXDSP Virtual Platform Test

The transmission rate test results of RapidIO and Ethernet system are shown in Table 1. The units of data and rate are byte and Gbps.

Table 1. RapidIO and Ethernet transmission rate test table.

RapidIO data	RapidIO rate	Ethernet data	Ethernet rate
40	0.640	84	1.344
56	0.896	128	2.048
132	2.112	288	4.608
198	3.168	520	8.320
276	4.416	784	12.544

The RapidIO transmission rates of the first and second data in Table 1 are 0.640 Gbps and 0.896 Gbps respectively, which can not make full use of bandwidth and channel resources. The Ethernet transmission rate in Table 1 is calculated with MAC frame data. Due to the extra field consumption, the data transmission rate can not reach the theoretical peak value of 16 Gbps.

4.2 MPIRIO Parallel Communication Interface Test

As shown in Table 2, the test results of data transmission under 8 and 16 HXDSP models are shown. Because only one process can be executed on each HXDSP chip, the process meaning in MPIRIO test is equal to HXDSP model.

Table 2. MPIRIO in 8 and 16 process data transmission test.

Function	Data (byte)	8 Thread (μs)	16 thread (μs)
MPIRIO_Send	10000	202	313
MPIRIO_Reduce	10000	431	650
MPIRIO_Gather	10000	393	635
MPIRIO_Bcast	10000	148	232
MPIRIO_Scatter	10000	176	252

The test results of convolution operation using MPIRIO are shown in Table 3. The input step size of convolution operation is 1. The test is carried out under four HXDSP simulators, and the convolution data are evenly distributed to each HXDSP simulator.

Table 3. MPIRIO convolution test table.

Input	Kernel	Output	Time (μs)
$33 \times 33 \times 3$	$2 \times 2 \times 3$	$32 \times 32 \times 1$	721
$34 \times 34 \times 3$	$3 \times 3 \times 3$	$32 \times 32 \times 1$	830
$35 \times 35 \times 3$	$4 \times 4 \times 3$	$32 \times 32 \times 1$	1015
$36 \times 36 \times 3$	$5 \times 5 \times 3$	$32 \times 32 \times 1$	1304

MPIRIO transforms the third convolution operation data in Table 3 into matrix multiplication of 16×48 and 48×1 , and uses basic partition multiplication to test the time. Therefore, designing convolution operation with MPIRIO is similar to designing matrix multiplication.

4.3 Convolution Operation Acceleration Test

Generally, the implementation of convolution operation is realized by using six layer for loop, tag with Algorithm 3.

Experiment with Algorithm 1 and Algorithm 3 to test the number of clock cycles and speedup ratio under different input channels, as shown in Table 4. The step size is 1 and the convolution kernel size is 3×3 .

With the increase of the number of input channels, the speedup ratio of Algorithm 1 relative to Algorithm 3 will gradually increase. The CPU of this paper is 4 cores, and the speedup ratio of HXDSP chip relative to CPU can reach $5.33\times$.

Table 4. Convolution speedup of Algorithms 1 and 3.

Input channel	Speedup	Algorithm3 (clocks)	Algorithm1 (clocks)
3	19.64	13824	704
16	21.24	73728	3472
64	21.3	294912	13840
128	21.32	589524	27664
256	21.32	1179648	55312

4.4 Meta-learning Optimization Mechanism Test

1. Training process of meta-learning optimizer

In this paper, two-layer LSTM network is the meta-learning optimizer to under the environment of CPU and Ubuntu. The optimized model is a simple four layer model, and the dataset is mnist. Cross entropy as the loss function.

Model training 200 epoch, batch_size set to 128, the optimizer runs 100 timesteps in each epoch and updates every 20 timesteps. The training process is shown in Fig. 4.

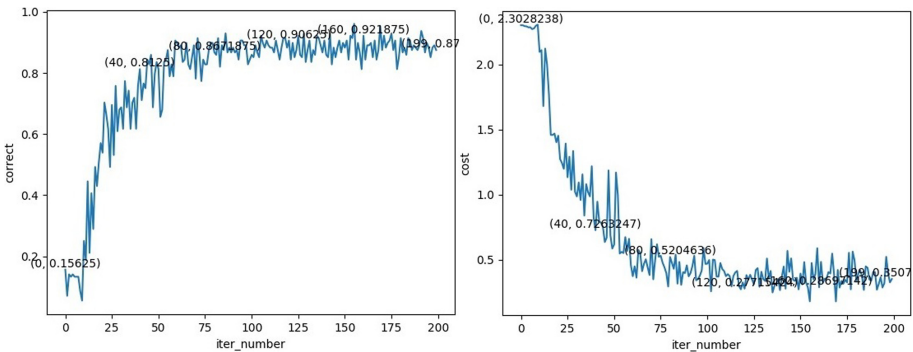


Fig. 4. Meta-learning optimizes training process on HXDSP virtual platform

On the left of Fig. 4, with the increase of iter_number, the accuracy of MNIST model is increasing. On the right of Fig. 4, the value of the loss function decreases gradually. This proves that the meta-learning optimization mechanism plays a role in the model training process on the virtual platform.

Set different epochs, steps and different optimizer expansion length to test the accuracy of MNIST model trained by meta-learning optimizer and the time of each epoch in the training process. The train results are shown in Table 5.

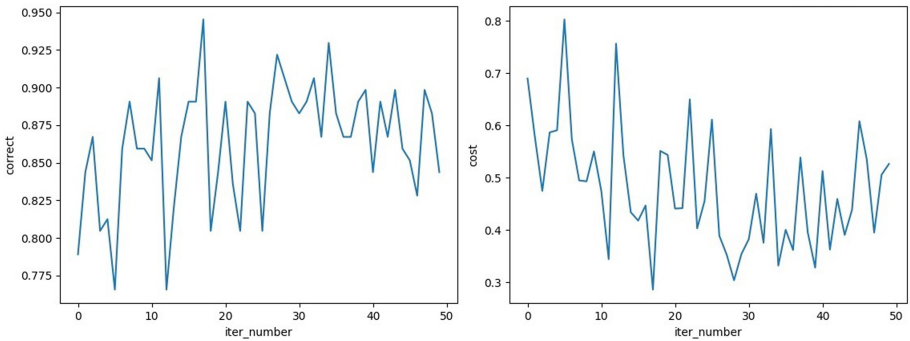
Through the comparison of different epochs, it can be found that when the epochs is about 200 and the expansion length is 20, the achievable accuracy rate is 93%, and the loss value is 0.25.

Table 5. Training data table of meta-learning.

Epochs	Steps	Unroll length	Loss	Accuracy
100	100	20	0.75	79%
200	200	20	0.25	92%
500	200	20	0.25	93%
200	200	40	0.26	92%

2. Optimization process of meta-learning optimizer

Host and virtual platform transmit data through shared memory, including loss and gradient information of MNIST model. Update the parameters of MNIST model. The test process is shown in Fig. 5.

**Fig. 5.** learning optimizes testing process on HXDSP virtual platform

On the left of Fig. 5, after the meta-learning optimizer initializes the model, the accuracy of the model can reach about 78%. During the running of the model, the meta-learning optimizer optimize the model on the virtual platform to gradually improve the accuracy of the model. On the right of Fig. 5, the loss of the model shows a downward trend. However, there is jitter during operation. This is because the input data is random, and the model may forget some information during adjustment.

5 Future Work

One of the future research directions is the mechanism of continuous learning. Although meta-learning optimization can adjust the parameters of the model, some information may be forgotten in the process of optimization. Combined with the continuous learning mechanism and the computing and storage resources based on the HXDSP many-core virtual platform, achieve faster and more accurate deep learning model optimization.

6 Conclusions

In this paper, we design a many-core virtual platform based on HXDSP processor. Using the parallel computing capability provided by HXDSP chip and the pipeline optimization mechanism provided by virtual platform, the running process of deep learning model is accelerated. Based on the heterogeneous computing system composed of CPU/HXDSP, the meta-learning optimization mechanism is realized. The experiment proves that HXDSP many-core virtual platform can realize the deployment and optimization of deep learning model.

Acknowledgments. This work was supported by the Core Electronic Devices, High-end Generic Chips and Basic Software of National Science and Technology Major Projects of China under Grant No. 2012ZX01034-001-001. And we thank the Anhui Province Key Laboratory of High Performance Computing at Hefei in UTSC for their support of our research.

References

1. Hao, X., Zhang, G., Ma, S.: Deep learning. *Int. J. Semant. Comput.* **10**(3), 417–439 (2016)
2. Li, T.-M., Gharbi, M., Adams, A., et al.: Differentiable programming for image processing and deep learning in halide. *ACM Trans. Graph.* **37**(4), 1–13 (2018)
3. Noda, K., Yamaguchi, Y., Nakadai, K., Okuno, H.G., Ogata, T.: Audio-visual speech recognition using deep learning. *Appl. Intell.* **42**(4), 722–737 (2014). <https://doi.org/10.1007/s10489-014-0629-7>
4. Lu, W., Zhou, Z., Zhang, L., Zheng, G.: Multi-target tracking by non-linear motion patterns based on hierarchical network flows. *Multimed. Syst.* **25**(4), 383–394 (2019). <https://doi.org/10.1007/s00530-019-00614-y>
5. Weng, Yu., Xia, C.: A new deep learning-based handwritten character recognition system on mobile computing devices. *Mob. Netw. Appl.* **25**(2), 402–411 (2019). <https://doi.org/10.1007/s11036-019-01243-5>
6. Kang, W., Chung, J.: Power- and time-aware deep learning inference for mobile embedded devices. *IEEE Access* **7**, 3778–3789 (2019)
7. Bhardwaj, K., Lin, C., Sartor, A., et al.: Memory- and communication-aware model compression for distributed deep learning inference on IoT. *ACM Trans. Embed. Comput. Syst.* **18**(5), 1–22 (2019)
8. Mayer, R., Jacobsen, H.-A.: Scalable deep learning on distributed infrastructures: challenges, techniques, and tools. *ACM Comput. Surv.* **53**(1), 1–37 (2020)
9. Liu, Y., Lang, W., Jia, G.: Realization and performance analysis of matrix multiplication on HXDSP platform. *Comput. Eng.* **45**(4), 25–29 (2019)
10. Shang, C., Yang, F., Huang, D., et al.: Data-driven soft sensor development based on deep learning technique. *J. Process Control* **24**(3), 223–233 (2014)
11. Shah, S.-I.-A., Khanvilkar, S., Khokhar, A.: RapidIO traffic management and flow arbitration protocol. *IEEE Commun. Mag.* **44**(7), 45–52 (2006)
12. Cossu, G., Sturniolo, A., Messa, A., et al.: Full-Fledged 10Base-T ethernet underwater optical wireless communication system. *IEEE J. Sel. Areas Commun.* **36**(1), 194–202 (2018)
13. Rivas-Gomez, S., Gioiosa, R., Peng, I.-B., et al.: MPI windows on storage for HPC applications. *Parallel Comput.* **77**(9), 38–56 (2018)

14. Schumacher, J., Hayley, K., Boutin, L.-C., et al.: PPAPI: a program for ground-water modeling tasks in distributed parallel computing environments. *J. Ground Water* **56**(2), 248–250 (2018)
15. Berg, R., König, L., Rūhaak, J., Lausen, R., Fischer, B.: Highly efficient image registration for embedded systems using a distributed multicore DSP architecture. *J. Real-Time Image Process.* **14**(2), 341–361 (2014). <https://doi.org/10.1007/s11554-014-0457-3>
16. Ma, Y., Suda, N., Cao, Y., et al.: ALAMO: FPGA acceleration of deep learning algorithms with a modularized RTL compiler. *Integration* **62**(6), 14–23 (2018)
17. Zhou, F., Wu, B., Li, Z.: Deep Meta-Learning: Learning to Learn in the Concept Space. arXiv preprint [arXiv:1802.03596](https://arxiv.org/abs/1802.03596) (2018)
18. Hong, G., Kang, S., Kim, C.-S., et al.: Efficient parallel join processing exploiting SIMD in multi-thread environments. *ICE Trans. Inf. Syst.* **101**(3), 659–667 (2018)
19. Qiu, K., Zhu, Y., Xu, Y., et al.: BRLoop: constructing balanced retimed loop to architect STT-RAM-based hybrid cache for VLIW processors. *Microelectron. J.* **83**(1), 137–146 (2019)
20. Chen, K., Tao, W.: Learning linear regression via single-convolutional layer for visual object tracking. *IEEE Trans. Multimed.* **21**(1), 86–97 (2018)
21. Shermin, T., Murshed, M., Lu, G., et al.: An Efficient Transfer Learning Technique by Using Final Fully-Connected Layer Output Features of Deep Networks. arXiv preprint [arXiv:1712.01252](https://arxiv.org/abs/1712.01252) (2018)
22. Zhou, Y., Zhang, M., Zhu, J., Zheng, R., Wu, Q.: A randomized block-coordinate adam online learning optimization algorithm. *Neural Comput. Appl.* **32**(16), 12671–12684 (2020). <https://doi.org/10.1007/s00521-020-04718-9>



Development of Low-Cost Indoor Positioning Using Mobile-UWB-Anchor-Configuration Approach

Ang Liu (✉), Shiwei Lin, Xiaoying Kong, Jack Wang, Gengfa Fang, and Yunlong Han

Faculty of Engineering and Information Technology,
University of Technology Sydney, Sydney, Australia
Ang.Liu@student.uts.edu.au

Abstract. In recent years, with the growth of indoor positioning demand, many kinds of indoor positioning technologies have been studied. Compared with other technologies, UWB indoor positioning technology has the advantages of high positioning accuracy and strong anti-interference ability. However, the high cost of UWB hardware limits the application of this technology to practical applications. In particular, the effective communication distance of the UWB is within 10 m, and if used in a large-area indoor environment, a plurality of anchor points is required to be installed to ensure the positioning accuracy. This leads to a high system hardware cost.

In this paper, we proposed a mobile-UWB-anchor-network approach. We changed the fixed anchors in the UWB system into moving anchors to reduce the number of anchors in the area and reduce the cost of the system. This new approach is verified using experiments.

Keywords: Indoor positioning · UWB moving anchors · Low-cost

1 Introduction

With the development of the internet of things technology, supply chain, and intelligent city, people's activities are more and more concentrated indoors. The traditional positioning technology in the outdoor environment, such as GPS, can no longer meet the positioning requirements in an indoor complex environment. However, the actual requirements of the intelligent warehouse, logistics monitoring, human capital monitoring, and so on, also make the research of indoor positioning technology become a hot spot.

UWB (Ultra-Wideband) has an extremely high bandwidth, operating frequency between 3.1 GHz and 10.6 GHz, does not occupy the existing bandwidth resources and does not interfere with existing bandwidth signals. At the same time, the UWB signal has nanosecond pulse width and good penetration ability. Because of these advantages of UWB, UWB can achieve centimeter-level positioning in an indoor environment, and the positioning accuracy can reach less than 10 cm under the condition of no occlusion [1]. On the other hand, the radiation of UWB signal is very low, only 1/1000 of the

radiation of mobile phone signal, which is much lower than that of Wi-Fi signal, so it will not interfere with the instrument and equipment, and can meet the indoor positioning needs of some special environments, such as factories and hospitals [2].

At present, there are many research directions and methods of indoor positioning technology, such as Bluetooth positioning technology, Wi-Fi positioning technology, and UWB positioning technology, all of which have their advantages and disadvantages. Kong et al. summarized that a navigation system needs to consider quality attributes which include accuracy, availability, reliability, robustness, safety, security, and response time; On the other hand, also need to consider development constraints which include maintainability, usability, development complexity, cost constraint, time constraint and client and supplier collaboration capacity [3]. One of the difficulties in the development of indoor positioning technology is how to measure the positioning accuracy and system cost.

For some positioning technologies, such as Bluetooth indoor positioning technology, Wi-Fi indoor positioning technology, the deployment cost is low, positioning accuracy is also relatively low. The positioning accuracy can only reach 2 m. Although the high precision technology such as UWB, can meet the centimeter-level positioning accuracy, the hardware cost is remarkably high, it is difficult to meet the large-scale application.

This paper will pay attention to this research question: how to use a more reasonable hardware configuration to reduce the system cost without affecting the accuracy of UWB indoor positioning. In this paper, we present a new indoor positioning approach to change the fixed-anchor-point in a conventional UWB positioning system into a mobile-anchor-point configuration to reduce the number of anchor points required in a large-area indoor environment, thereby reducing system cost. This approach is verified using the design and experiments at the positioning accuracy level of the mobile anchor point system.

This paper is organized into the following sections. Section 2 reviews the literature. Section 2 presents the UWB indoor positioning system hardware and configuration.

2 Literature Review

Indoor positioning has two sides: indoor positioning mechanisms and indoor positioning technologies. Indoor positioning mechanisms are general positioning algorithms that geometrically locate the position of an object. Indoor positioning technologies are using sensors and other hardware related technologies to obtain positioning-related measurements to feed into indoor positioning algorithms. In this section, we will review the current literature on these 2 sides.

2.1 Indoor Positioning Algorithm

Fingerprint

The fingerprint algorithm first measures the signal strength of each position in the area and builds a database of the measurement results. When locating, it needs to measure the signal intensity of the target point. By comparing the previous fingerprint database

which includes RSSI (received signal strength indication) and the corresponding position coordinates, the location of the target point can be determined.

The fingerprint algorithm can be divided into two steps. The first step is called the off-line phase. The main purpose of the off-line phase is to build a database. The second part, called the on-line phase, is to get the location information by comparing the database information [4]. Fingerprint algorithm also has some disadvantages, first, it needs to collect a large amount of data when building a database. If it is applied in a large regional environment, it will be a lot of work in building such a database [5]. Secondly, after the establishment of the database in the first step, if the indoor facilities or arrangements change, the signal strength of each point will be affected, which will also lead to the localization error.

TOA (Time of Arrival)

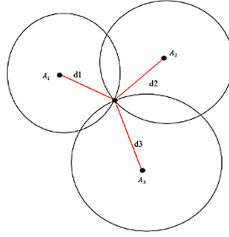


Fig. 1. Time of arrival

TOA is a method to locate the target by measuring the distance [6]. The principle of this method is to calculate the distance between the target point and the anchor point by measuring the arrival time of the signal and then calculate the Tag coordinates by the distance. The positioning in a 2D plane requires at least 3 anchors. After measuring the distance between the target label and three different anchors, the exact position of the target label can be obtained by a series of geometric algorithms [7]. The working principle of TOA will be explained by the mathematical formula below.

As shown in Fig. 1, the coordinates at the target point of the positioning system is $tagT = (x, y, z)$, anchors coordinates are $A_i = (x_i, y_i, z_i)$ $i = 1, 2, 3 \dots N$.

$$d_i = (t_i - t_0) * c \quad (1)$$

where d_i : Distance between anchor i and tag, t_i : Signal arrival time, t_0 : Signal sending time, c : speed of light

$$\begin{aligned} d_1^2 &= (x - x_1)^2 + (y - y_1)^2 + (z - z_1)^2 \\ d_2^2 &= (x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 \\ d_3^2 &= (x - x_3)^2 + (y - y_3)^2 + (z - z_3)^2 \\ d_4^2 &= (x - x_4)^2 + (y - y_4)^2 + (z - z_4)^2 \end{aligned} \quad (2)$$

From the above mathematical formula, the following results can be obtained

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2)2(z_1 - z_2) \\ 2(x_1 - x_3) & 2(y_1 - y_3)2(z_1 - z_3) \\ 2(x_1 - x_4) & 2(y_1 - y_4)2(z_1 - z_4) \end{bmatrix}^{-1} * \begin{bmatrix} d_2^2 - d_1^2 - x_2^2 + x_1^2 - y_2^2 + y_1^2 - z_2^2 + z_1^2 \\ d_3^2 - d_1^2 - x_3^2 + x_1^2 - y_3^2 + y_1^2 - z_3^2 + z_1^2 \\ d_4^2 - d_1^2 - x_4^2 + x_1^2 - y_4^2 + y_1^2 - z_4^2 + z_1^2 \end{bmatrix} \quad (3)$$

TOA algorithm requires extremely high time accuracy and time synchronization between anchor and tag. Even if the time error is small, for example, the time error of 1ns will produce a distance error of 0.3 m when multiplied by the speed of light.

Therefore, TDOA (Time Difference of Arrival) which is easier to implement is obtained based on TOA.

TDOA (Time Difference of Arrival)

The basic principle of TDOA is to make use of the characteristics of hyperbolic, that is, the difference between the distance from the point on the hyperbola to the two focal points is a fixed value [8]. Figure 2 shows the schematic diagram of the TDOA.

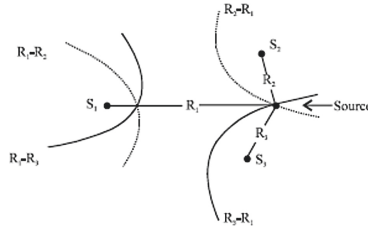


Fig. 2. Time difference of arrival

The basic principle of TDOA is to locate the transmission time delay difference between Tag and two anchors. As shown above, S_1 , S_2 , S_3 represent three anchors, R_1 , R_2 , R_3 represent the distance between Tag and the corresponding anchor point, respectively. If the distance difference between Anchor 1 and Anchor 2 is R_{21} , then the tag must be on a hyperbola that takes S_1 and S_2 as the focus. By the same token, Tag will also be on hyperbolic curves with S_1 and S_3 as the focus. The coordinates of the tag are obtained by calculating the intersection of the two hyperbolic curves.

The advantage of TDOA over TOA is that when calculating the distance between anchors, the initial time t_0 is eliminated by subtraction.

Therefore, TDOA does not need time synchronization between tag and anchor, but only needs to synchronize between base stations, which greatly reduces the technical difficulty.

2.2 Indoor Positioning Technology

Wi-Fi

Due to the development and popularization of the Internet, the Wi-Fi network has been well constructed. At the same time, the Wi-Fi network construction has the advantages of low cost, easy deployment, and the Wi-Fi network has been widely used to cover the indoor environment as a way for people to visit the Internet in a daily way [8]. As an indoor positioning technology, Wi-Fi mainly uses the fingerprint location method based on RSSI. This method completes the location through off-line sampling and online positioning. It also has some shortcomings of fingerprint location, such as the offline phase is very time-consuming [9]. If the indoor parameters change, the fingerprint database needs to be re-established [10].

The advantages of this method are the hardware cost is low, the transmission rate is high, but the transmission distance is short, and the power consumption is high. It can achieve meter level positioning.

Bluetooth

Bluetooth is a low-power wireless transmission technology. By installing Bluetooth access points indoors, multiple users can connect at the same time and determine the location of access devices [11]. Bluetooth positioning technology is generally suitable for small indoor positioning, such as a single room warehouse. The biggest advantage of Bluetooth technology is that the device size is small, the power consumption is low, and it is easy to be integrated into mobile terminals such as mobile phones. However, Bluetooth technology also has some shortcomings, such as poor stability in a complex indoor environment, easy to be interfered with, and expensive equipment [12].

When Bluetooth technology is used for indoor positioning, the commonly used method is also fingerprint location algorithm based on RSSI, and the accuracy can reach about two meters under ideal conditions. Through some auxiliary methods, such as integration of the weighted average algorithm, inertial navigation algorithm, Kalman filter algorithm, and so on, the positioning accuracy can be further improved to decimeter level [12].

UWB (Ultra-wideband)

The UWB (Ultra-wideband) has an exceedingly high bandwidth, the operating frequency band is 3.1–10.6 GHz, and the duration is noticeably short. Therefore, the theoretical positioning accuracy can reach centimeter-level [13]. The transmission distance of UWB is generally in the range of 10 m, and a large amount of data can be transmitted in a short period. Because the UWB uses truly short pulses and the transmission power is exceedingly small, it is very suitable for real-time indoor positioning [14]. The system communicates with unknown nodes entering the communication range by pre-arranged fixed-position anchor nodes and then determines the position information by triangulation or fingerprint algorithm.

UWB has many advantages, such as low power consumption, strong anti-jamming ability, strong penetration, can be used in non-line-of-sight, will not interfere with other equipment, it is also safe for people [15].

Our research will focus on UWB technologies. UWB system can be divided into three layers, management layer, service layer, and site layer. The focus of our approach is the site layer, the site layer is mainly composed of anchor points and positioning tags. The tag is associated with the target and broadcasts its location to the anchor. The anchor receives the location information of the tag and then passes the information to the position calculation engine. After receiving the information uploaded by the anchor points, the calculation engine calculates the exact position of the target tag by the corresponding algorithm.

3 Mobile-UWB-Anchor-Network Configuration Approach

In the conventional UWB positioning approach, a large amount of fixed position anchors is deployed. Figure 3 illustrates this positioning approach.

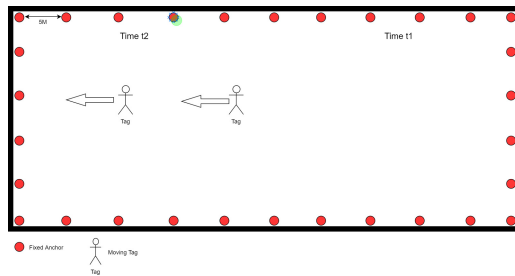


Fig. 3. Conventional UWB anchor network using fixed anchors

If tags are moving from Time t1 to Time t2 and leave the space where these tags were in time t1 without tags, the anchors in the time t1 area are not used for positioning.

We propose a new approach to reduce the deployment of such a large number of fixed anchors (Fig. 4).

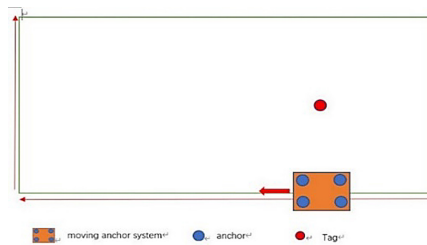


Fig. 4. New mobile-anchor-network approach

We adopted a new way of thinking about the system of placing the originally fixed anchor on a mobile platform to form a mobile anchor point system. This mobile anchor system moves at a constant speed along a fixed track within a certain area. When tags need to be positioned appear in the region, the mobile platform first determines its position based on displacement time, velocity, direction. If there are at least three anchor points to communicate with Tag, the position of the Tag can be calculated by the TOA method, thus achieving the positioning of the mobile anchor system. Here is the experimental flowchart (Fig. 5):

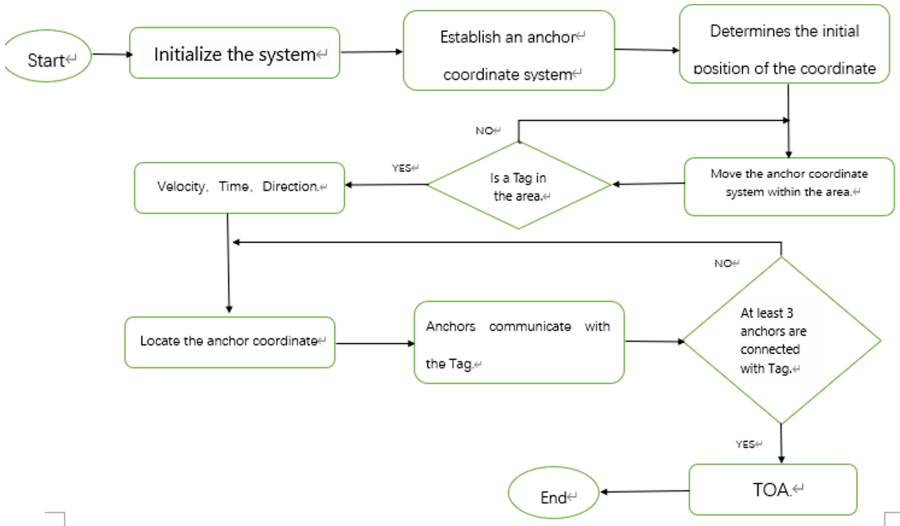


Fig. 5. Mobile anchor system experimental flowchart.

For the positioning algorithm TOA, because the actual experimental state signal will be interfered with by noise and cause the measurement distance error, so the ranging error ε_i is added to the experimental simulation. The actual range should be:

$$d_i = r_i - \varepsilon_i \tag{4}$$

Where i is the number of anchors ($i = 1, 2, 3, 4$); d_i is an actual range between anchor i and Tag; r_i is measuring range between anchor i and Tag; ε_i is ranging error. Assuming:

$$K_i = x_i^2 + y_i^2 + z_i^2 \tag{5}$$

Where (x_i, y_i, z_i) is coordinate of anchor i ($i = 1, 2, 3, 4$).

From Eqs. (3), (4), and (5), the following equation can be obtained:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 2(x_1 - x_2) & 2(y_1 - y_2)2(z_1 - z_2) \\ 2(x_1 - x_3) & 2(y_1 - y_3)2(z_1 - z_3) \\ 2(x_1 - x_4)2(y_1 - y_4)2(z_1 - z_4) \end{bmatrix}^{-1} \quad (6)$$

$$* \begin{bmatrix} r_2^2 - r_1^2 + K_1 - K_2 + \varepsilon_2^2 + 2r_1\varepsilon_1 - 2r_2\varepsilon_2 - \varepsilon_1^2 \\ r_3^2 - r_1^2 + K_1 - K_3 + \varepsilon_3^2 + 2r_1\varepsilon_1 - 2r_3\varepsilon_3 - \varepsilon_1^2 \\ r_4^2 - r_1^2 + K_1 - K_4 + \varepsilon_4^2 + 2r_1\varepsilon_1 - 2r_4\varepsilon_4 - \varepsilon_1^2 \end{bmatrix}$$

Where (x, y, z) is the coordinate point of Tag
Assuming:

$$D = \begin{bmatrix} (x_1 - x_2) & (y_1 - y_2) & (z_1 - z_2) \\ (x_1 - x_3) & (y_1 - y_3) & (z_1 - z_3) \\ (x_1 - x_4) & (y_1 - y_4) & (z_1 - z_4) \end{bmatrix} \quad (7)$$

$$X = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad (8)$$

$$c = \begin{bmatrix} r_2^2 - r_1^2 + K_1 - K_2 \\ r_3^2 - r_1^2 + K_1 - K_3 \\ r_4^2 - r_1^2 + K_1 - K_4 \end{bmatrix} \quad (9)$$

$$\Delta = \begin{bmatrix} \varepsilon_2^2 + 2r_1\varepsilon_1 - 2r_2\varepsilon_2 - \varepsilon_1^2 \\ \varepsilon_3^2 + 2r_1\varepsilon_1 - 2r_3\varepsilon_3 - \varepsilon_1^2 \\ \varepsilon_4^2 + 2r_1\varepsilon_1 - 2r_4\varepsilon_4 - \varepsilon_1^2 \end{bmatrix} \quad (10)$$

Substituting Eq. (7) (8) (9) (10) into Eq. (6), we can get:

$$2DX = c + \Delta \quad (11)$$

From the least square method, the Tag coordinate can be obtained as:

$$X = \frac{1}{2} (D^T D)^{-1} D^T (c + \Delta) \quad (12)$$

The main purpose of this experiment is to verify whether the moving anchor points can accurately locate Tag after the fixed anchor points are turned into mobile anchor points. The cost of the UWB anchors is high. Using this approach, we reduce the cost of UWB hardware. Our approach can be verified using experiments. The next section will discuss the verification results.

4 Experiment

4.1 Experiment Hardware

This section will introduce the verification results. We use experiments and simulation to analyze this approach.

Experiments are set up using DWM1001 UWB equipment. See Fig. 6.



Fig. 6. DWM1001-DEV development boards

For an indoor positioning system, positioning accuracy and system cost are two especially important factors. We have carried out some tests on the DWM1001 board, and installed four fixed anchors and a moving tag, to compare the data collected by the tag with the real position. It can be determined that the positioning accuracy of the UWB system can reach centimeter-level within the effective communication distance. The biggest problem with using UWB to build an indoor positioning system is the high cost. Because the effective communication range of UWB is generally less than 10m, if UWB indoor positioning is used in a large indoor environment such as an airport or warehouse in the future, hundreds or even thousands of fixed anchors will need to be installed, which will lead to too high the cost of the system. Therefore, in our experiments, we focus on how to reduce the cost of the UWB indoor positioning system and achieve high precision positioning at the same time.

4.2 Experimental Results

The overall idea of this experiment is to reduce the number of anchors by changing the fixed anchor into a moving anchor to reduce the cost of the system.

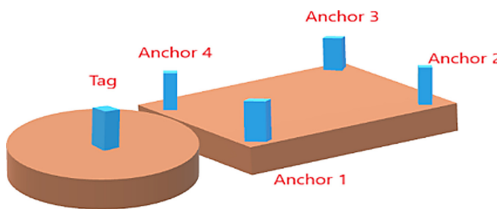


Fig. 7. Mobile-Anchor UWB system

As shown in Fig. 7, four UWB anchors are fixed to the four corners of one box, a tag is fixed on a robot and the computer is connected to the tag to read data, then the box and robot are connected, thus building a mobile UWB positioning system.

The coordinate information of the anchor is shown in the following Fig. 8:

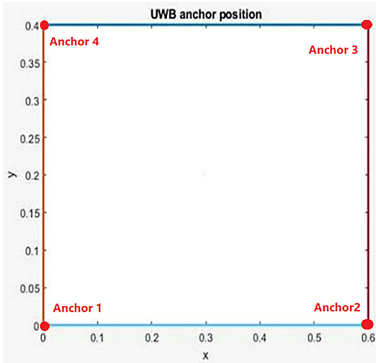


Fig. 8. Coordinate of anchor



Fig. 9. System movement route

Figure 9 shows the experimental site and the moving trail of the UWB system. The experimental site is in the common area of UTS building 11 level 12, and the moving trail of the whole system is shown in the figure, from point A to point B then to point C, where the trajectory AB is X-axis and BC is Y-axis. Some of the data is shown in the following table (Table 1).

Table 1. Experiment 1 data

Time(s)	0	0.06	0.16	0.28	0.66	0.96	1.06	1.26	1.36
x	0.023	0.005	0.016	0.005	0.021	0.02	0.035	0.034	0.039
y	0.034	0.015	0.031	0.041	0.048	0.04	0.058	0.055	0.061
z	0.043	0.044	0.055	0.054	0.065	0.055	0.075	0.059	0.068

From data that was read out from the tag, we can get the information of four anchors, including the anchor ID, coordinate position of anchors, and the distance between the anchor point and Tag. Combining these data, we can analyze the measurement data as shown in Fig. 10. The following trajectory diagram can be obtained utilizing the data analysis of MATLAB.

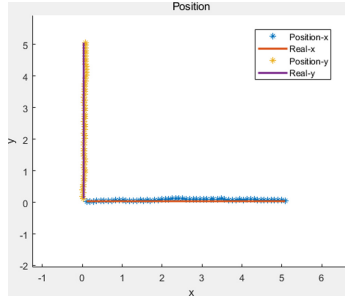


Fig. 10. Experiment trajectory diagram (Color figure online)

The red line in Fig. 10 represents the trajectory of the anchor and the real trajectory of the whole system; the blue and yellow points represent the trajectory of the Tag located through the anchor. The mobile UWB anchor positioning system moves 5 m along the X-axis and the Y-axis. From the above figure, it can be found that the system can accurately locate the tag position during the movement. The specific positioning error is represented by the following two figures:

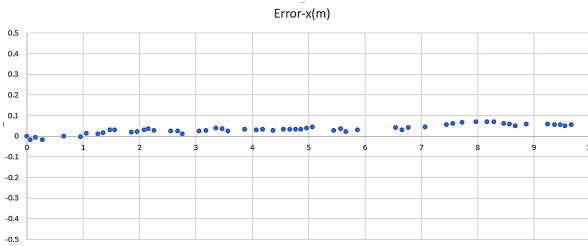


Fig. 11. Tag position during the system moves along the X-axis

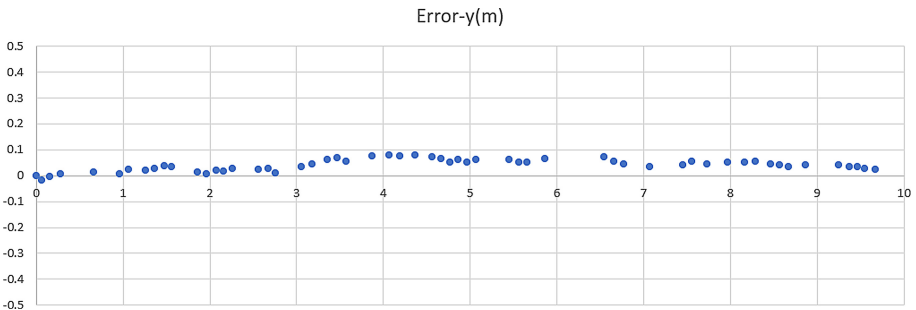


Fig. 12. Tag position during the system moves along the Y-axis

Table 2. Data analysis

	AVG	MAX	RMS
X-Error	0.033193	0.07	0.039183
Y-Error	0.039596	0.08	0.045534

Figure 11 and Fig. 12 respectively show the result of tag positioning during the period when the mobile UWB anchor positioning system moves along the X-axis and the Y-axis. From the above two figures (Figs. 11 and 12) and table (Table 2), it can be found that when the system moves along the X-axis, the tag's positioning error is 7 cm, the average error is 3.32 cm, and when the system moves along the Y-axis, the tag's positioning error is 8 cm, the average error is 4 cm. Therefore, the above results can show that the mobile UWB anchor positioning system can accurately locate the tag when the system moving on a fixed track, and it also verifies the feasibility of the method of reducing the cost by turning the fixed anchor point of the UWB into a mobile anchor point.

5 Conclusion and Future Work

This article introduces several mainstream indoor positioning methods and algorithms in detail and proposes a new mobile anchor UWB positioning method for the high hardware cost of the UWB positioning system which can reduce the usage of anchor points in the same area. Through experiments, the positioning error of the mobile anchor UWB point positioning system moving along a fixed track is about 10 cm, which verifies that this method is feasible. In future work, we will coordinate different positioning methods, such as inertial navigation, lidar, or visual SLAM, with the UWB mobile anchor positioning system, so that the mobile positioning system can no longer move along a fixed track, also it can move to the area that needs positioning more flexibly.

References

1. Yanfeng, L., Zhuo, T., Cunbao, S., Weibin, S.: Fast and resource efficient method for indoor localization based on fingerprint with varied scales. *J. Commun.* **40**, 172–179 (2019)
2. Vinicchayakul, W., Promwong, S., Supanakoon, P.: Study of UWB indoor localization using fingerprinting technique with different number of antennas. In: *International Computer Science and Engineering Conference (ICSEC)*, Chiangmai, pp. 1–4. IEEE (2016)
3. Kong, X., Liu, L., Tran, T.P., Sandrasegaran, K.: Analysis of stakeholder concerns for vehicle navigation system architecture solution. In: *IEEE Fifth International Conference on Communications and Electronics (ICCE)*, Thomson Reuters, Da Nang (2014)
4. Vinicchayakul, W., Promwong, S.: Improvement of fingerprinting technique for UWB indoor localization. In: *The 4th Joint International Conference on Information and Communication Technology, Electronic and Electrical Engineering (JICTEE)*, Chiang Rai, pp. 1–5. IEEE (2014)

5. Liu, H., Darabi, H., Banerjee, P., Liu, J.: Survey of wireless indoor positioning techniques and systems. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **37**(6), 1067–1080 (2007)
6. Spano, D., Ricciato, F.: Opportunistic time-of-arrival localization in fully asynchronous wireless networks. *Pervasive Mob. Comput.* **37**, 139–153 (2017)
7. Krishnan, S., Sharma, P., Guoping, Z., Woon, O.H.: A UWB based localization system for indoor robot navigation. In: *IEEE International Conference on Ultra-Wideband*, Singapore, pp. 77–82. IEEE (2007)
8. Kaune, R.: Accuracy studies for TDOA and TOA localization. In: *15th International Conference on Information Fusion*, Singapore, pp. 408–415. IEEE (2012)
9. Xujian, H., Hao, W.: WIFI indoor positioning algorithm based on improved Kalman filtering. In: *International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, Changsha, pp. 349–352. IEEE (2016)
10. Chen, C.-S.: Hybrid TOA/AOA geometrical positioning schemes using linear lines of position for mobile location. *IEICE Trans. Fundam. Electronics Commun. Comput. Sci.* **98**(8), 1676–1679 (2015)
11. Feldmann, S., Kyamakya, K., Zapater, A., Lue, Z.: An indoor bluetooth-based positioning system: concept, implementation and experimental evaluation. In: *International Conference on Wireless Networks*, 2003, CSREA, Las Vegas, vol. 272 (2003)
12. Wang, Q., Guo, Y., Yang, L., Tian, M.: An indoor positioning system based on ibeacon. In: Pan, Z., Cheok, A David, Müller, W., Zhang, M. (eds.) *Transactions on Edutainment XIII*. LNCS, vol. 10092, pp. 262–272. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54395-5_22
13. Gigl, T., Janssen, G.J., Dizdarevic, V., Witrisal, K., Irahauten, Z.: Analysis of a UWB indoor positioning system based on received signal strength. In: *4th Workshop on Positioning, Navigation and Communication*, Hannover, pp. 97–101. IEEE (2007)
14. Sahinoglu, Z., Gezici, S., Guvenc, I.: *Ultra-Wideband Positioning Systems*. Cambridge, New York (2008)
15. Alavi, B., Pahlavan, K.: Modeling of the TOA-based distance measurement error using UWB indoor radio measurements. *IEEE Commun. Lett.* **10**(4), 275–277 (2006)



BERT-CoQAC: BERT-Based Conversational Question Answering in Context

Munazza Zaib^{1(✉)}, Dai Hoang Tran¹, Subhash Sagar¹, Adnan Mahmood¹,
Wei E. Zhang^{1,2}, and Quan Z. Sheng¹

¹ Department of Computing, Macquarie University, Sydney 2109, Australia
munazza-zaib.ghori@hdr.mq.edu.au

² School of Computer Science, The University of Adelaide, Adelaide, Australia

Abstract. As one promising way to inquire about any particular information through a dialog with the bot, question answering dialog systems have gained increasing research interests recently. Designing interactive QA systems has always been a challenging task in natural language processing and used as a benchmark to evaluate machine's ability of natural language understanding. However, such systems often struggle when the question answering is carried out in multiple turns by the users to seek more information based on what they have already learned, thus, giving rise to another complicated form called *Conversational Question Answering (CQA)*. CQA systems are often criticized for not understanding or utilizing the previous context of the conversation when answering the questions. To address the research gap, in this paper, we explore how to integrate the conversational history into the neural machine comprehension system. On one hand, we introduce a framework based on publicly available pre-trained language model called BERT for incorporating history turns into the system. On the other hand, we propose a history selection mechanism that selects the turns that are relevant and contributes the most to answer the current question. Experimentation results revealed that our framework is comparable in performance with the state-of-the-art models on the QuAC (<http://quac.ai/>) leader board. We also conduct a number of experiments to show the side effects of using entire context information which brings unnecessary information and noise signals resulting in a decline in the model's performance.

Keywords: Machine comprehension · Information retrieval · Deep learning · Deep learning applications

1 Introduction

The field of conversational AI can be divided in to three categories namely, *goal-oriented dialogue systems*, *chat-oriented dialogue systems*, and *question answering (QA) dialogue systems*. The former two have been very researched upon topics, resulting in a number of successful dialogue agents such as Amazon Alexa,

Apple Siri and Microsoft Cortana. However, QA dialogue systems are fairly new and still require extensive research. To facilitate the growth of this category, many question answering challenges were proposed [1–3] which later gave rise to the field of *Conversational Question Answering* (CQA). CQA has introduced a new dimension of dialogue systems that combines the elements of both chat and question answering. Conversational QA is a “user ask, system respond” kind of setting where a user starts the conversation with a particular question or information need and the system searches its database to find an appropriate solution of that query. This could turn into a multi-turn conversation if the user needs to have more detailed information about the topic. The ability to take into account previous utterances is key to building interactive QA dialogue systems that can keep conversations active and useful. Yet, modeling conversation history in an effective way is still an open challenge in such systems.

Existing approaches have tried to address the problem of history conversation modeling by prepending history questions and answers to the current question and source passage [4]. Though this seems to be a simple method to improve the answer’s accuracy, in reality it fails to do so. Another approach used complex Graph Neural Networks [5] to deal with this issue. One recent work [6] introduced the use of history answer embeddings but they did not consider using relevant context rather than used entire history turns to find the answer span. Also, they did not draw the complete picture of the context to the model by eliminating history questions. Table 1 shows a chunk of dialogue extracted from the QuAC [7] dataset. In order to answer the query Q2, we expect the system to have the knowledge of Q1 and A1, so that it can easily decipher the “he” entity in Q2. This shows that modeling complete history is necessary when designing an effective conversational QA system. To address this shortcoming, in this paper, we emphasize the following research questions:

Table 1. A chunk of a dialogue from QuAC dataset.

Topic: Formative years and life’s calling		
ID	Role	Conversation
Q1	Usr	When was Kurien born?
A1	Sys	He was born on 26 November, 1921
Q2	Usr	Where was he born?
A2	Sys	Calicut, Madras Presidency (now Kozhikode, Kerala) in a Syrian Christian family

Research Question 1: *How can we utilize the context in an efficient way?*

To answer this, we propose an effective framework, called **B**idirectional **E**ncoder **R**epresentations from **T**ransformers based **C**onversational **Q**uestion **A**nswering in **C**ontext (BERT-CoQAC), that uses a mechanism to only extract the context

that is relevant to the current question by calculating the similarity score between the two. By doing so, our model would be able to generate better contextualized representation of the words, thus resulting in the better answer spans. Apart from this, we also propose the use of history questions to support and improve the effect of history answer embeddings when looking for an answer span.

Research Question 2: *What is the effect of incorporating the entire context into conversational question answering system?* To answer this, we perform extensive experiments and finds out that using the entire context results in decreasing the model’s performance due to the presence of unnecessary information in the provided input.

Thus, the contributions of our work are summarized as the following: i) we introduce a new method of incorporating selected history questions along with answer embeddings to model the complete conversation history; ii) we present that noise in context utterances could result in decline in model’s performance; iii) the experimental results shows that our method achieves better performance in accuracy than the other different state-of-the-art published models.

2 Related Work

The concept of BERT-CoQAC is similar to machine comprehension and can be termed as conversational machine comprehension (CMC). The difference between MC and CMC is that questions in MC are independent of each other whereas questions in CMC form a series of questions that requires a proper modeling of the conversation history in order to comprehend the context of current question correctly. Different models and approaches have been proposed to handle the complete conversation. [8] used hierarchical models, first capturing the meaning of individual utterances and then combining them as discourses. [9] extended this concept with the attention mechanism to attend to significant parts of the utterances on both word and utterance level, respectively. In another study [10], a systematic comparison between hierarchical and non-hierarchical methods was conducted and the authors proposed a variant that weighs the context with respect to context-query relevance.

High quality conversational dataset such as QuAC [7] and CoQA [4] have provided the researchers a great source to work deeply in the field of CMC. In our work, we choose to work on QuAC because it encourages users to participate more in the information seeking dialogue. In this setting, the information seeker has the access only to the title of the paragraph and can pose free-form questions to learn about the hidden text of Wikipedia paragraph.

The baseline model for QuAC is based on single turn machine comprehension model known as BiDAF [11] which was further extended to BiDAF++ w/xctx [7] that marked the history answers in the source paragraph. BiDAF++ was further improved in BERT-HAE [6], where the concept of history answer

embeddings was used. These embeddings were then concatenated with the given passage to identify the answer span. FlowQA [12] introduced a mechanism to grasp the history of conversation by generating an intermediate representation of the previously carried out conversation. Later, Graphflow [5] introduced a graph neural network (GNN) based model to construct the context-aware graph and capture the flow of the conversation.

The recent state-of-the-art advancements in pre-trained language models such as BERT, ELMo, and GPT-2 have led to the rapid proliferation of research interests in the field of machine reading comprehension. These models can be utilized effectively in tasks with small datasets as the relationship between the words are already learnt by these models during the pre-training phase. Out of all the pre-trained language models, BERT is known to produce state-of-the-art results in MRC tasks.

Just after a short period after its introduction, BERT has been widely used on different conversational datasets such as learning dialog context representations on MultiWOZ¹ or evaluating the open-domain dialog system on DailyDialog² dataset. On the QuAC leaderboard³, various approaches [6, 13] use BERT as a base model for conversational question answering. The difference between aforementioned approaches and our model is that they either capture none or all of the previous conversational history turns whereas our proposed BERT-CoQAC finds relevant history turns in order to predict the accurate answer.

3 Methodology

In the following section, we present our model that addresses the two research questions described in Sect. 1.

3.1 Task Formulation

Given a source paragraph P and a question Q , the task is to find an answer A to the question provided the context and can be formulated as follows:

Input: The input of BERT-CoQAC consists of current question Q_i , given passage P , history questions Q_{i-1}, \dots, Q_{i-k} and the embeddings of history answers $HE_{i-1}, \dots, HE_{i-k}$,

Output: The answer A_i identified using start span and end span generated by the model.

where i and k represents the indices of turn and the number of dialogue history considered, respectively.

3.2 BERT-CoQAC

Figure 1 represents the overall framework of BERT-CoQAC, which consists of:

¹ <http://dialogue.mi.eng.cam.ac.uk/index.php/corpus/>.

² <http://yanran.li/dailydialog.html>.

³ <https://quac.ai/>.

History Selection Module that calculates the context-query relevance score and selects the history turns that are expected to be more relevant to the question, and

History Modeling Module that takes the selected history turns along with the current question and passage, and transforms them into the format required by BERT.

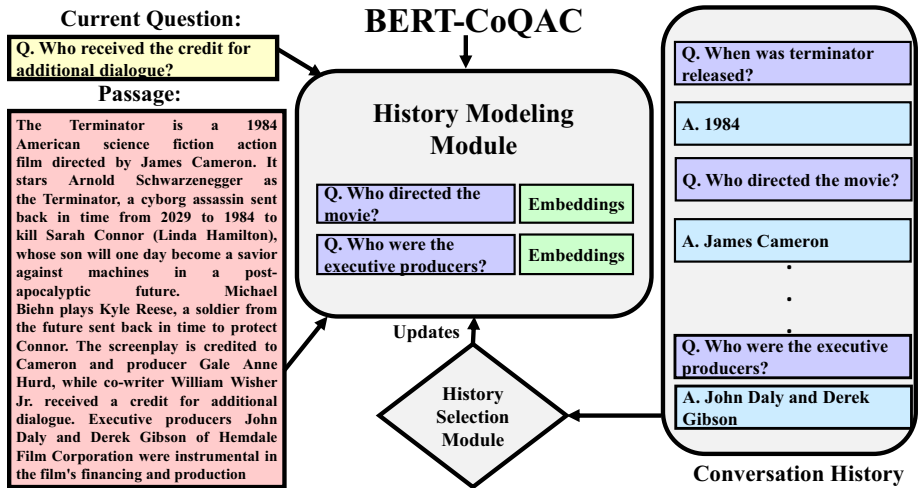


Fig. 1. Modular representation of BERT-CoQAC. It shows the input formulation and the components of our model.

Inside the *History Modeling* module, we leverage the strengths of pre-trained language model and adapt BERT to suit our task’s requirements. Figure 2 illustrates our model’s architecture that includes embeddings of history answers along with the history questions. Another factor worth noting here is that *History Selection* module only selects those history questions and answer embeddings that have cosine similarity score greater than the threshold value calculated using:

$$score_i = \frac{h_i \cdot q}{\|h_i\| \cdot \|q\|} \quad (1)$$

where h_i denotes current history turn and q represents the query. Finally, the scores are normalized using softmax function to get the probability distribution as shown in Eq. 2.

$$p_i = \frac{\exp(score_i)}{\sum_{j=0}^n \exp(score_j)} \quad (2)$$

We performed different experiments and found out that turns having score greater than or equal to 0.5 contributes more in generating accurate answer

span, thus, the threshold value is set to 0.5. Turns having value less than this will be filtered out. Our model feeds the conversation history to BERT in a natural way and generates two types of tokens for history answer embeddings. E_H/E_N shows that whether the token is a part of history answer or not. These embeddings have influence on the information that the other tokens possess. The history questions are not part of the passage, so we cannot “embed” them directly to the input sequences. Instead, we prepend the historical questions in the same sequence as that of embeddings to improve the answer span. Let T_i be the BERT-representation of the i^{th} token and S be the start vector. The probability of this token being the start token is $P_i = \frac{e^{S.T_i}}{\sum_k e^{S.T_k}}$. The probability of a token being the end token is computed likewise. The loss is the average of the cross entropy loss for the start and end positions.

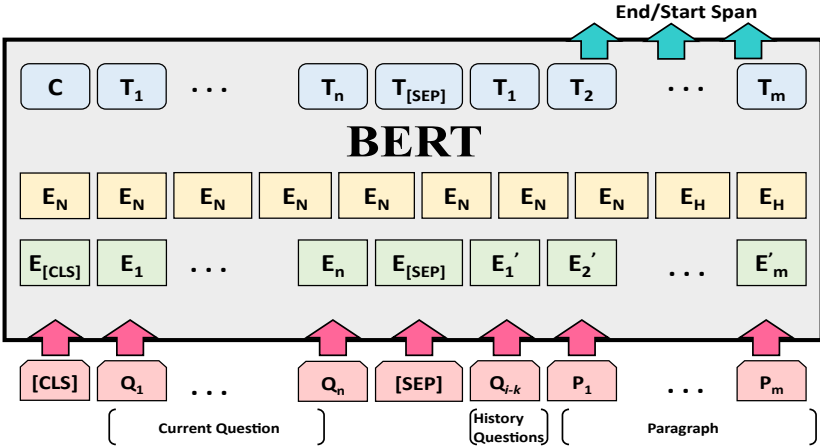


Fig. 2. Architecture of BERT-CoQAC model. History questions are prepended with the passage and E_N/E_H denotes whether the token is present in history or not.

4 Experimental Setup

In this section, we describe the setup of our experiments for the evaluation of the proposed BERT-CoQAC model, including the dataset, model training, the comparison baseline methods, implementation details, and the evaluation metrics.

4.1 The Dataset

The motivation behind QuAC⁴ (Question Answering in Context) comes from the idea of teacher-student setup where a student asks a series of questions about

⁴ <http://quac.ai/>.

a topic to get in-depth information about it. The issue in this scenario is that most of the questions are context-dependant, can be abstract, and might not have any answer. It is up to the teacher to utilize and shift all the knowledge they have to provide the best answer. The seeker has access only to the heading of the given paragraph and the answers are provided by generating the start and end span in the paragraph. The training/validations sets have 11,000/1,000 questions across 14,000 dialogues. Every dialogue can have a maximum of 12 dialogue turns, which constitutes 11 history turns at most.

4.2 Model Training

$(Q_i, Q_i^{k.P}, HA_i^k, A_i)$ denotes a single instance of training where $Q_i^{k.P}$ denotes the paragraph prepended with history questions in the same order as that of history answers, HA . This instance is first transformed into example variation where each variation has only one history turn from the conversation history. A context-query relevance based history selection module then considers k relevant history turns. A new instance, $(Q_i, Q_i^{k.P}, AE_i^k, A_i)'$, is formed by merging all the selected variations and used as an input to the BERT-CoQAC model. Since the length of the passages is greater than the maximum sequence length, therefore, we use the sliding window approach to split lengthy passages as suggested in the BERT [14] model.

4.3 Comparison Systems

A brief description of competing methods is as follows:

- **BiDAF++** [11]: BiDAF++ extends BiDAF by introducing contextualized embeddings and self-attention mechanism in the model.
- **BiDAF++ w/ 2-ctx** [7]: It takes 2 history turns into account when extending BiDAF++. It also concatenates the marker embeddings to passage embeddings and adds dialogue turn number into question embeddings.
- **FlowQA** [12]: FlowQA introduces a mechanism that incorporates intermediate representations generated during the process of answering previous questions to make the model be able to grasp the latent semantics of the history.
- **BERT-HAE** [6]: This model is built on pre-trained language model, BERT, to model history conversation using history answer embeddings.

Our proposed BERT-CoQAC framework is an improved model based on BERT-HAE with the introduction of history selection mechanism along with the history questions to improve the model’s accuracy.

4.4 Hyper-Parameter Settings

The model is implemented using Tensorflow and uses version v0.2 of QuAC dataset. We utilize the BERT-Base model (uncased) having maximum length of the sequence set to 384. Document stride is 128 and the maximum answer length

is of 40. The batch size is set to 12. The turns to be incorporated are selected on the basis of their relevance to the current question. The optimizer is Adam with a learning rate of $3e-5$. The number of training epochs is 3. We use two NVIDIA Tesla P100 16 GB GPUs.

4.5 Evaluation Metrics

For the evaluation purpose, we use not only the F_1 score but also the human equivalence score for questions (HEQ-Q) and dialogues (HEQ-D) [7]. **HEQ-Q** represents the percentage of exceeding or matching the human performance on questions and **HEQ-D** represents the percentage of exceeding or matching the human performance on dialogues.

5 Evaluation Results

Table 2 and Table 3 shows the evaluation results of our model on the QuAC dataset. Our model outperforms the baseline methods and BERT-HAE model on all the three metrics i.e. F_1 , HEQ-Q, and HEQ-D and answer our research questions as follows.

Research Question 1: *How can we utilize the context in an efficient way?*

From Table 2, we can make the following observations. i) Using conversation history has a significant effect when answering the current question. This holds true for both BiDAF++ and BERT-based methods. ii) Incorporating relevant history turns rather than the entire conversation has a significant effect when answering the current question. Our experiments confirms the hypothesis and outperforms the competing methods. iii) Apart from history answer embeddings, history questions also plays a significant part in improving answer’s accuracy. The effect of including complete history turn (consisting of both question and answer) is presented more clearly in the next paragraph. iv) BERT-CoQAC with simple experiment setup performs just as good as FlowQA that uses convoluted mechanisms to model the history. v) The training time of our model is way more efficient than that of FlowQA and is slightly better than BERT-HAE as well which proves the efficiency of our proposed architecture.

Research Question 2: *What is the effect of incorporating the entire context into conversational question answering system?*

We conducted extensive experiments without using context-query relevance mechanism and took maximum (i.e. 11) number of turns into consideration. Table 3 shows that selection of relevant history is essential to generate better answer spans. Utilizing entire context results in decreasing the model’s performance. However, our model still performs better than BERT-HAE just by simply adding the history turns, comprising of both previous questions and answers, into the model. Our model provides the highest accuracy after introducing 5 history turns which is an improvement on BERT-HAE model that provides high accuracy after 6 turns which shows that introducing complete history is necessary for better answer accuracy rather than just using history answer embeddings.

Table 2. The evaluation results based on val/test scores. The top section is the baseline methods, the middle section is BERT-HAE with different methods and the bottom section lists the best performing models.

Models	F1	HEQ-Q	HEQ-D	Train time
BiDAF++	51.8/50.2	45.3/43.3	2.0/2.2	–
BiDAF++ w/2Ctx	60.6/60.1	55.7/54.8	5.3/4.0	–
BERT + PHA	61.8/–	57.5/–	4.7/–	7.2
BERT + PHQA	62.0/–	57.5/–	5.4/–	7.9
BERT + HAE	63.1/62.4	58.6/57.8	6.0/5.1	10.1
BERT-CoQAC	65.1/64.0	60.2/59.6	6.6/5.8	8.9
FlowQA	–/64.1	–/59.6	–/5.8	56.8

Table 3. The evaluation results of BERT-CoQAC with the varying number of turns on QuAC dataset.

Evaluation with 11 history turns on QuAC			
History turns	F1	HEQ-Q	HEQ-D
1	61.57	57.58	4.7
2	63.04	58.9	6.0
3	62.58	58.64	5.4
4	62.46	58.04	5.4
5	63.4	58.86	6.3
6	62.73	58.39	5.8
7	62.94	58.89	6.2
8	62.16	58.10	4.6
9	62.9	58.05	5.6
10	62.23	58.26	5.7
11	62.13	58.11	5.6

6 Conclusion and Future Work

This paper is an effort to introduce a BERT-based framework, BERT-CoQAC, for effective conversational question answering in context. The proposed framework first selects the relevant history turns using the context-query relevance and models the history conversation by adding the history questions along with the embeddings to generate better context of the conversation. To verify our hypothesis, we conduct a number of experiments to analyze the effectiveness of relevant conversational turns on the overall accuracy of the model using a real-world dataset. The results show the effectiveness of our model.

The history selection mechanism used in the model is efficient but comprises of a very basic strategy. This paper is a work in progress and we plan on improving the history selection strategy as future work.

References

1. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: SQuAD: 100,000+ questions for machine comprehension of text. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, pp. 2383–2392. Association for Computational Linguistics, November 2016
2. Joshi, M., Choi, E., Weld, D., Zettlemoyer, L.: TriviaQA: a large scale distantly supervised challenge dataset for reading comprehension. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada, pp. 1601–1611. Association for Computational Linguistics, July 2017
3. Kočiský, T., et al.: The NarrativeQA reading comprehension challenge. *Trans. Assoc. Comput. Linguist.* **6**, 317–328 (2018)
4. Reddy, S., Chen, D., Manning, C.D.: CoQA: a conversational question answering challenge. *TACL Trans. Assoc. Comput. Linguist.* **7**, 249–266 (2019)
5. Chen, Y., Wu, L., Zaki, M.J.: GraphFlow: exploiting conversation flow with graph neural networks for conversational machine comprehension. *CoRR*, abs/1908.00059 (2019)
6. Qu, C., Yang, L., Qiu, M., Bruce Croft, W., Zhang, Y., Iyyer, M.M.: BERT with history answer embedding for conversational question answering. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, 21–25 July 2019, pp. 1133–1136 (2019)
7. Choi, E., et al.: QuAC: question answering in context. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2174–2184, October–November 2018
8. Serban, I.V., Sordoni, A., Bengio, Y., Courville, A.C., Pineau, J.: Building end-to-end dialogue systems using generative hierarchical neural network models. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA, 12–17 February 2016, pp. 3776–3784 (2016)
9. Xing, C., Wu, Y., Wu, W., Huang, Y., Zhou, M.: Hierarchical recurrent attention network for response generation. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI 2018), the 30th innovative Applications of Artificial Intelligence (IAAI 2018), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI 2018), New Orleans, Louisiana, USA, 2–7 February 2018, pp. 5610–5617 (2018)
10. Tian, Z., Yan, R., Mou, L., Song, Y., Feng, Y., Zhao, D.: How to make context more useful? An empirical study on context-aware neural conversational models. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, 30 July–4 August, Volume 2: Short Papers, pp. 231–236 (2017)
11. Seo, M.J., Kembhavi, A., Farhadi, A., Hajishirzi, H.: Bidirectional attention flow for machine comprehension. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017, Conference Track Proceedings (2017)

12. Huang, H.-Y., Choi, E., Yih, W.: FlowQa: grasping flow in history for conversational machine comprehension. CoRR, abs/1810.06683 (2018)
13. Yeh, Y.T., Chen, Y.-N.: FlowDelta: modeling flow information gain in reasoning for conversational machine comprehension. CoRR, abs/1908.05117 (2019)
14. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, 2–7 June 2019, Volume 1 (Long and Short Papers), pp. 4171–4186 (2019)



Streaming Algorithms for Monotone DR-Submodular Maximization Under a Knapsack Constraint on the Integer Lattice

Jingjing Tan¹, Dongmei Zhang²(✉), Hongyang Zhang³, and Zhenning Zhang³

¹ School of Mathematics and Information Science, Weifang University,
Weifang 261061, People's Republic of China

tanjingjing1108@163.com

² School of Computer Science and Technology, Shandong Jianzhu University,
Jinan 250101, People's Republic of China

zhangdongmei@sdjzu.edu.cn

³ Department of Operations Research and Information Engineering, Beijing
University of Technology, Beijing 100124, People's Republic of China

zhanghongyang@emails.bjut.edu.cn, zhangzhenning@bjut.edu.cn

Abstract. Maximization of non-negative monotone submodular set functions under a knapsack constraint have been extensively studied in the last decade. Here, we consider the streaming algorithms of this problem on the integer lattice, or on a multi-set equivalently. This is more realistic for many practical problems such as sensor location and influence maximization. It is well known that submodularity and diminishing return submodularity are not equivalent on the integer lattice. We mainly focus on maximizing the diminishing return submodular (DR-submodular) functions with knapsack constraint on the integer lattice. Finally, by utilizing the binary search algorithm as a subroutine, we design an online streaming algorithm called DynamicMRT. Furthermore, we prove that it is a $(1/3 - \varepsilon)$ -approximation algorithm with $O(K \log K)$ memory complexity and $O(\log K)$ query complexity per element.

Keywords: Streaming algorithm · DR-submodular · Integer lattice · Knapsack constraint

1 Introduction

Submodular maximization is a classical and widely studied problem in both combinatorial optimization and machine learning. Up to now, there exist many elegant approximation algorithms, such as greedy algorithms and local search algorithms [3, 4, 6, 8, 9, 11, 12, 16]. With the development of science and technology, data streaming of massive data is produced every second from social networks, financial markets, sensor networks data, etc. In many real-world applications, the main memory capacity of individual computers is much less than the amount of

input data. We need to process data in a streaming fashion for such a large-scale case. That is, we assume the item in the ground set $E = \{e_1, e_2, \dots, e_n\}$ arrive sequentially, and we must make a decision for the current element before the arrival of the next one, which means we cannot use the classical submodular maximization algorithms. One of the classical design techniques for this situation is streaming algorithm. Traditionally, streaming algorithms are generally measured by approximation ratio, running time and memory complexity. As the running time of a streaming algorithm depends on the number of oracle queries required for processing every element in the worse case, We just need to calculate the number of queries to analyze the time complexity. Moreover, the number of passes reading all over the data is also an important parameter for evaluating a streaming algorithm. It is obvious that we can get higher accuracy and better performance by running multiple passes. But for most applications, it is not realistic to visit data stream for more than once. Many researches are focus on one-pass algorithms with high accuracy.

Badanidiyuru et al. [1] devised the first $(1/2 - \varepsilon)$ -approximation Sieve-Streaming algorithm with memory complexity $O(K \log K)$ for submodular maximization with a cardinality constraint. Buchbinder et al. [2] obtained a streaming algorithm with a constant $(1/4)$ -approximation and improved the memory complexity to $O(K)$. Norouzi-Fard et al. [18] proved that any streaming algorithm with memory complexity $O(n/K)$ does not admit an approximation ratio better than $1/2$ unless $P = NP$. For streaming algorithms of maximizing traditional submodular function with a knapsack constraint, Wolsey [24] first obtained an algorithm with $(1 - 1/e^\beta) \approx 0.35$ -approximation, where β is the unique root of the equation $e^x = 2 - x$. Yu et al. [26] obtained a single-pass $(1/3 - \varepsilon)$ -approximation and Huang et al. [13] gave a $(0.4 - \varepsilon)$ -approximation algorithm for the general case with knapsack constraint.

Most conclusions about the problem of maximization submodular functions consider submodular functions are set-submodular functions. The input is a subset of a ground set and the output is a real value [2, 5, 7, 10, 14, 17, 19, 25, 27]. However, in many practical scenarios, it is more nature to consider submodular functions over a multi-set or equivalently, submodular function over the integer lattices \mathbb{N}^E for some finite set E . We call a function $f : \mathbb{N}^E \rightarrow \mathbb{R}^+$ is diminishing return submodular(DR-submodular) if $f(\mathbf{x} + \chi_{e_i}) - f(\mathbf{x}) \geq f(\mathbf{y} + \chi_{e_i}) - f(\mathbf{y})$ for arbitrary $\mathbf{x} \leq \mathbf{y}$ and $e_i \in E$, where χ_{e_i} is the i -th unit vector. Most of maximization DR-submodular functions usually appear in submodular welfare problem [15, 20] and the budget allocation problem [21, 22]. Soma et al. [23] designed a polynomial-time approximation algorithm for maximizing monotone DR-submodular functions under cardinality constraints, polymatroid constraints and knapsack constraints. In this paper, we design a streaming algorithm for submodular maximization subject to a knapsack constraint over the integer lattice. For a given $\mathbf{x} \in \mathbb{N}^E$, it is a n -dimensional vector where E is the ground set of size n , $f : \mathbb{N}^E \rightarrow \mathbb{R}^+$ is a monotone DR-submodular function defined on the integer lattice. Therefore, the problem is defined as

$$\text{maximize } f(\mathbf{x}) \text{ subject to } \mathbf{c}^T \mathbf{x} \leq K, \quad (1)$$

where $\mathbf{c} : \mathbb{N}^E \rightarrow \mathbb{R}_+$, $\mathbf{c}(e)$ is the weight of element e , and $K \in \mathbb{N}$ is the total budget.

First, we design a streaming algorithm with the known optimal value of problem (1). Obviously, the OPT is unknown. To address this case, we introduce a two pass streaming algorithm. In the first round, we get the maximum value of standard unit vectors. By utilizing the value, we get the estimation interval of the OPT. In order to decreasing the pass of algorithm, we have to lower the condition of calculating the maximum value of the standard unit vector over the whole stream. For each coming element e , we update the maximum value of the standard unit vector along with the arrived elements. Based on such operations, we design a DynamicMRT algorithm with one pass. We also prove that such an algorithm is $(1/3 - \varepsilon)$ -approximation streaming algorithm with a per-element query number $O((\log^2 K)/\varepsilon)$, and the space complexity is $O(K \log K/\varepsilon)$.

The paper builds up as follows. In Sect. 2, we introduce some definitions and lemmas. In Sect. 3, we propose a streaming algorithm for DR-submodular maximization problem and the algorithm analysis. Finally, in Sect. 4, we conclude our work.

2 Preliminaries

In this section, we list some definitions and lemmas which will be used later. We first give some notations.

Throughout this paper, $E = \{e_1, e_2, \dots, e_n\}$ denotes the ground set of size n . For a positive integer $k \in \mathbb{N}$, $[k]$ denotes the set $\{1, 2, \dots, k\}$. $\mathbf{x} \in \mathbb{N}^E$ is a n -dimensional vector where $\mathbf{x}(e_i)$ is the component of coordinate $e_i \in E$ of \mathbf{x} . Let $\mathbf{0}$ denotes the zero vector, χ_{e_i} denotes the standard unit vector with 1 at the i -th component and 0 otherwise, χ_X denotes the characteristic vector of $X \subseteq E$, and $\mathbf{x}(X) := \sum_{e_i \in X} \mathbf{x}(e_i)$.

For $\mathbf{x} \in \mathbb{N}^E$, $\text{supp}^+(\mathbf{x}) = \{e \in E | \mathbf{x}(e) > 0\}$. Let $\{\mathbf{x}\}$ denote the multi-set where the element e appears $\mathbf{x}(e)$ times, and $|\{\mathbf{x}\}| := \mathbf{x}(E)$. Let $\mathbf{x} \vee \mathbf{y}$ be the coordinate wise maximum of \mathbf{x} and \mathbf{y} , and $\mathbf{x} \wedge \mathbf{y}$ denotes the coordinate wise minimum. For each element $e \in E$, $(\mathbf{x} \vee \mathbf{y})(e) = \max\{\mathbf{x}(e), \mathbf{y}(e)\}$ and $(\mathbf{x} \wedge \mathbf{y})(e) = \min\{\mathbf{x}(e), \mathbf{y}(e)\}$. We define $\{\mathbf{x}\} \setminus \{\mathbf{y}\} := \{(\mathbf{x} \setminus \mathbf{y}) \vee \mathbf{0}\}$ for arbitrary two multi-sets \mathbf{x} and \mathbf{y} .

Let $f : \mathbb{N}^E \rightarrow \mathbb{R}_+$ be a function defined on the integer lattice. We say that f is monotone non-decreasing if $f(\mathbf{x}) \leq f(\mathbf{y})$ for any $\mathbf{x} \leq \mathbf{y}$ and f is nonnegative and normalized if $f(\mathbf{x}) \geq 0$ for any $\mathbf{x} \in \mathbb{N}^E$ and $f(\mathbf{0}) = 0$.

In the following article, we recall the definitions of (lattice) submodularity and diminishing return submodularity of the function f on the integer lattice.

Definition 2.1. A function $f : \mathbb{N}^E \rightarrow \mathbb{R}_+$ is (lattice) submodular, if it satisfies

$$f(\mathbf{x}) + f(\mathbf{y}) \geq f(\mathbf{x} \vee \mathbf{y}) + f(\mathbf{x} \wedge \mathbf{y}),$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{N}^E$.

Definition 2.2. A function $f : \mathbb{N}^E \rightarrow \mathbb{R}_+$ is diminishing return submodular (DR-submodular), if it satisfies

$$f(\mathbf{x} + \chi_e) - f(\mathbf{x}) \geq f(\mathbf{y} + \chi_e) - f(\mathbf{y}),$$

for any $e \in E, \mathbf{x}, \mathbf{y} \in \mathbb{N}^E$ with $\mathbf{x} \leq \mathbf{y}$.

From the definition of lattice submodularity and diminishing return submodularity of f , we can see the lattice submodularity is a weaker condition than the diminishing return submodularity when the domain is the integer lattice.

Denote by \mathcal{F}_b the set of all non-negative monotone and DR-submodular functions with $f(\mathbf{0})$ and domain $\{\mathbf{x} \in \mathbb{N}^E : \mathbf{x} \leq \mathbf{b}\}$. For $f \in \mathcal{F}_b$, and vectors $\mathbf{x}, \mathbf{y} \in \mathbb{N}^E$, we use the notation $f(\mathbf{x}|\mathbf{y}) = f(\mathbf{x} + \mathbf{y}) - f(\mathbf{y})$ to denote the marginal increment of a vector \mathbf{x} with respect to a vector \mathbf{y} . Let \mathbf{x}^* and OPT be an optimal solution vector and the optimal value of the problem (1), respectively.

3 Streaming Algorithm for DR-Submodular Maximization

In this section, we design a $(1/3 - \varepsilon)$ -approximation streaming algorithm for DR-submodular maximization under a knapsack constraint on the integer lattice. This algorithm will be incorporated into other algorithms introduced later.

3.1 A Streaming Algorithm with Known Optimal Value

In this subsection, we give a MarginalRatio Thresholding (α, v) algorithm under the assumption that the optimal value of the objective function is known. We can see the detailed description in Algorithm 1.

Algorithm 1. MarginalRatio Thresholding (α, v) with known optimal value

Input: $f \in \mathcal{F}_b, \mathbf{c} \in \mathbb{N}^E, E, \alpha \in (0, 1]$, and $v : \alpha OPT \leq v \leq OPT$.

output: a vector $\mathbf{x} \in \mathbb{N}^E$.

```

1:  $\mathbf{x} \leftarrow \mathbf{0}$ ;
2: for  $i = 1, 2, \dots, n$  do
3:   if  $\mathbf{c}^T \mathbf{x} < K$  then
4:      $l \leftarrow \mathbf{Binary\ Search}\left(f, \mathbf{x}, \mathbf{b}, \mathbf{c}, e, K, \frac{\alpha v - f(\mathbf{x})}{K - \mathbf{c}^T \mathbf{x}}\right)$ ;
5:     if  $\mathbf{c}^T(\mathbf{x} + l\chi_{e_i}) \leq K$  then
6:        $\mathbf{x} \leftarrow \mathbf{x} + l\chi_{e_i}$ ;
7:   end
8: end
9: return  $\mathbf{x}$ 

```

Throughout this subsection, we denote $\tilde{\mathbf{x}}$ as the output of Marginal Ratio Thresholding (α, v) .

Lemma 3.1. Let \mathbf{x}_{i-1} be the initial vector of the i th iteration, and e_i be the element exactly arriving at the i th iteration of Algorithm 1. The amount l_i returns from Algorithm 1. Then, for each iteration i of Algorithm 1, we have

$$f(\mathbf{x}_{i-1} + l_i \chi_{e_i}) \geq \frac{\alpha v \mathbf{c}^T \mathbf{x}_i}{K}. \quad (2)$$

Lemma 3.2. $f(\chi_e | \mathbf{x}^*) < \frac{\alpha v c(e)}{K}$, $e \in \{\mathbf{x}^*\} \setminus \{\tilde{\mathbf{x}}\}$, where $\tilde{\mathbf{x}}$ is the final solution vector returned by Algorithm 1 with $\mathbf{c}^T \mathbf{x} < K$ and \mathbf{x}^* is an optimal solution.

There are two cases that an element $e \in \{\mathbf{x}^*\}$ is not added to the last vector $\{\tilde{\mathbf{x}}\}$. One is that e does not pass the condition of the threshold value, another is that its addition would breach the knapsack constraint. We name the later case as follows:

Algorithm 2. Binary Search ($f, \mathbf{x}, \mathbf{b}, \mathbf{c}, e, \tau$)

Input: $f \in \mathcal{F}_b$, $e \in E$, $\mathbf{c}, \mathbf{x} \in \mathbb{N}^E$, and $\tau \in \mathbb{R}_+$.

Output: $l \in \mathbb{N}$.

```

1:  $l_l \leftarrow 1, l_r \leftarrow \left\lfloor \frac{\mathbf{b}(e) - \mathbf{c}^T \mathbf{x}}{\mathbf{c}(e)} \right\rfloor$ ;
2: if  $\frac{f(l_r \chi_e | \mathbf{x})}{l_r \mathbf{c}(e)} \geq \tau$  then
3:   return  $l_r$ 
4: end
5: if  $f(\chi_e | \mathbf{x}) < \tau$  then
6:   return 0
7: end
8: while  $l_l < l_r + 1$ , do
9:    $m = \lfloor \frac{l_l + l_r}{2} \rfloor$ 
10:  if  $\frac{f(m \chi_e | \mathbf{x})}{m \mathbf{c}(e)} \geq \tau$  then
11:     $l_l = m$ ,
12:  else
13:     $l_r = m$ ,
14:  end
15: end
16: return  $l_l$ 

```

Definition 3.1. An element $e \in \{\mathbf{x}^*\}$ is called bad if e satisfied the condition of the threshold value but the total size exceeds K when added, i.e., $f(l_e \chi_e | \mathbf{x}) \geq \frac{\alpha v - f(\mathbf{x})}{K - \mathbf{c}^T \mathbf{x}}$, $\mathbf{c}^T(\mathbf{x} + l_e \chi_e) > K$ and $\mathbf{c}^T \mathbf{x} \leq K$, where \mathbf{x} is the initial vector as e arrives.

Lemma 3.3. If $v \leq f(\mathbf{x}^*)$ and there have been no bad item, then $f(\tilde{\mathbf{x}}) \geq (1 - \alpha)v$ holds.

Algorithm 3. Singleton

```

1: procedure Singleton;
2:    $\mathbf{x} \leftarrow \mathbf{0}$ .
3:   while item  $e$  is arriving do
4:     if  $f(\mathbf{b}(e)\chi_e) > f(\mathbf{x})$  then  $\mathbf{x} \leftarrow \mathbf{b}(e)\chi_e$ .
5:   return  $\mathbf{x}$ .

```

Algorithm 3 returns the best singleton of the data. Let $\tilde{\mathbf{x}}'$ be the output of Algorithm 3. If some $e \in \{\mathbf{x}^*\}$ is bad, then combining with $\tilde{\mathbf{x}}'$, we can obtain a $(1/3 - \varepsilon)$ approximation.

Theorem 3.1. Let MarginalRatio-Thresholding($2/3, v$) (Algorithm 1) and Singleton algorithm (Algorithm 3) run in parallel. Thus approximation ratio must be better than $(1/3 - \varepsilon)$. The combining algorithm with the known OPT requires one pass with space complexity $O(K)$, and a per-element query number $O(\log K)$.

Proof. If there exists no bad item, we know $f(\tilde{\mathbf{x}}) \geq (1 - \alpha)v$ from Lemma 3.3. Now, we assume that we have a bad item $e \in E$. Let \mathbf{x}_e be the vector just before e arrives in MarginalRatio Thresholding. Then,

$$f(\mathbf{x}_e + l_e\chi_e) \geq \frac{\alpha v \mathbf{c}^T(\mathbf{x}_e + l_e\chi_e)}{K}.$$

Since $\mathbf{c}^T(\mathbf{x}_e + l_e\chi_e) > K$, then we obtain $f(\mathbf{x}_e + l_e\chi_e) \geq \alpha v$. Moreover, by DR-submodularity, we have

$$f(\mathbf{x}_e + l_e\chi_e) \leq f(\mathbf{x}_e) + f(l_e\chi_e).$$

Thus, either $f(\mathbf{x}_e)$ or $f(l_e\chi_e)$ is at least $(\alpha v)/2$. We have

$$f(\tilde{\mathbf{x}}) \geq f(\mathbf{x}_e) \geq \frac{\alpha v}{2}$$

or

$$f(\tilde{\mathbf{x}}') \geq f(l_e\chi_e) \geq \frac{\alpha v}{2}.$$

Then, We obtain $\max\{f(\tilde{\mathbf{x}}), f(\tilde{\mathbf{x}}')\} \geq \min\{\alpha/2, 1 - \alpha\}$, where the right-hand side is maximized to $1/3$ when $\alpha = 2/3$. So, if $v \in \mathbb{R}^+$ with $v \leq f(\text{OPT}) \leq (1 + \varepsilon)v$, we run MarginalRatio-Thresholding ($2/3, v$) and Singleton() in parallel and choose the better output that has the approximation ratio of $\frac{1}{3(1 + \varepsilon)} \geq 1/3 - \varepsilon$. Clearly, the space complexity of the algorithm is $O(K)$.

3.2 Streaming Algorithm for DR-Submodular Maximization

MarginalRatio Thresholding algorithm requires a good approximation of OPT. Generally, the OPT is unknown. However, we can use a traditional method to estimate the OPT. Based on the observation that $\max_{e \in E} f(\chi_e) \leq \text{OPT} \leq K \max_{e \in E} f(\chi_e)$, if $m = \max_{e \in E} f(\chi_e)$ is provided, there exists a value $v \in \mathbb{R}^+$ with $v \leq \text{OPT} \leq (1 + \varepsilon)v$ for $\varepsilon \in (0, 1]$ in the guess set $\mathcal{I} = \{(1 + \varepsilon)^s | m/(1 + \varepsilon) \leq (1 + \varepsilon)^s \leq Km/\alpha\}$. For each $v \in \mathcal{I}$, we can run Algorithm 4 in parallel and choose the best output.

Algorithm 4. MarginalRatio Thresholding(ε, α)**Input:** $f \in \mathcal{F}_b$, $\mathbf{c} \in \mathbb{N}^E$, E , $\varepsilon > 0$, $\alpha \in (0, 1]$.**output:** a vector $\mathbf{x} \in \mathbb{N}^E$.1: $m \leftarrow \max_{e \in E} f(\chi_e)$;2: $\mathcal{I} = \{(1 + \varepsilon)^s | m/(1 + \varepsilon) \leq (1 + \varepsilon)^s \leq Km/\alpha\}$;3: For each $v \in \mathcal{I}_\varepsilon$, set $\mathbf{x}^v \leftarrow \mathbf{0}$;4: **for** $i = 1, 2, \dots, n$ **do**5: **if** $\mathbf{c}^T \mathbf{x} < K$ **then**6: $l \leftarrow \text{Binary Search} \left(f, \mathbf{x}, \mathbf{b}, \mathbf{c}, e, K, \frac{\alpha v - f(\mathbf{x})}{K - \mathbf{c}^T \mathbf{x}} \right)$;7: **if** $\mathbf{c}^T (\mathbf{x} + l\chi_{e_i}) \leq K$ **then**8: $\mathbf{x} \leftarrow \mathbf{x} + l\chi_{e_i}$;9: **end**10: **end**11: **return** \mathbf{x}

Lemma 3.4 Algorithm 4 scans the data stream two passes, deserves space complexity at most $O(K \log K/\varepsilon)$, and a per-element query number $O(K \log K/\varepsilon)$.

Theorem 3.2 Combining MarginalRatio-Thresholding ($\varepsilon, 2/3$) and Singleton Algorithm, we return the better one as the output. The combining algorithm has a $(1/3 - \varepsilon)$ -approximation ratio.

As the proof is similar to Theorem 3.1, we omit it. In order to get one pass algorithm, we provide a DynamicMRT (ε, α) algorithm, which dynamically updates $m = \max_{e \in E} f(\chi_e)$ accompanying with the arriving elements. Therefore, the estimation interval $[m/(1 + \varepsilon), Km/\alpha]$ of the OPT is updating with the arriving elements. The details are presented in Algorithm 5. The main results is as follows.

Theorem 3.3. By running DynamicMRT($\varepsilon, 2/3$) and Singleton in parallel and outputting the better solution, we gain a $(1/3 - \varepsilon)$ -approximation streaming algorithm with single one pass, where $\varepsilon \in (0, 1]$. The space complexity is $O(K \log K/\varepsilon)$, and the query complexity is $O(\log^2 K/\varepsilon)$ per-element.

Proof. Suppose that $e \in E$ is an arriving element. We state that e always fails the condition of threshold value in DynamicMRT(ε, α) when $v > Km/\alpha$ (for $\alpha = 2/3$). If $v > Km/\alpha$, we can obtain

$$\begin{aligned} f(\mathbf{x} + l_e \chi_e) &\geq n \frac{\alpha v \mathbf{c}^T (\mathbf{x} + l_e \chi_e)}{K} \\ &> m \mathbf{c}^T (\mathbf{x} + l_e \chi_e) \\ &\geq |\{\mathbf{x} + l_e \chi_e\}| \max_{e' \in \{\mathbf{x} + l_e \chi_e\}} f(\chi_{e'}), \end{aligned} \quad (3)$$

where l_e returns from Algorithm 2, \mathbf{x} is the initial vector of this iteration. The last inequality of (3) holds from the fact that $\mathbf{c}(e) \geq 1$ and $m \geq$

Algorithm 5. DynamicMRT(ε, α)

Input: $f \in \mathcal{F}_b$, $\mathbf{c} \in \mathbb{N}^E$, E , $\varepsilon > 0$, $\alpha \in (0, 1]$.
output: a vector $\mathbf{x} \in \mathbb{N}^E$.

- 1: $\mathcal{I}_\varepsilon = \{(1 + \varepsilon)^s \mid s \in \mathbb{Z}\}$;
- 2: For each $v \in \mathcal{I}_\varepsilon$, set $\mathbf{x}^v \leftarrow \mathbf{0}$;
- 3: $m \leftarrow 0$;
- 4: **for** $i = 1, 2, \dots, n$ **do**
- 5: $m \leftarrow \max_{e \in E} \{m, f(\chi_e)\}$;
- 6: $\mathcal{I}^i = \{(1 + \varepsilon)^s \mid m/(1 + \varepsilon) \leq (1 + \varepsilon)^s \leq Km/\alpha\}$;
- 7: Delete all \mathbf{x}^v , where $v \notin \mathcal{I}^i$;
- 8: **for** $v \in \mathcal{I}^i$ **do**
- 9: **if** $\mathbf{c}^T \mathbf{x}^v < K$ **then**
- 10: $l \leftarrow \text{Binary Search} \left(f, \mathbf{x}, \mathbf{b}, \mathbf{c}, e, K, \frac{\alpha v - f(\mathbf{x})}{K - \mathbf{c}^T \mathbf{x}} \right)$;
- 11: **if** $\mathbf{c}^T (\mathbf{x}^v + l\chi_{e_i}) \leq K$ **then**
- 12: $\mathbf{x}^v \leftarrow \mathbf{x}^v + l\chi_{e_i}$;
- 13: **end**
- 14: **end**
- 15: **end**
- 16: **return** $\arg \max_{v \in \mathcal{I}^n} f(\mathbf{x}^v)$

$\max_{e' \in \{\mathbf{x} + l_e \chi_e\}} f(\chi_{e'})$. On the other hand, from $\mathbf{x} + l_e \chi_e = \sum_{e' \in \{\mathbf{x} + l_e \chi_e\}} \chi_{e'}$ and DR-submodularity, we have

$$\begin{aligned} f(\mathbf{x} + l_e \chi_e) &\leq f \left(\sum_{e' \in \{\mathbf{x} + l_e \chi_e\}} \chi_{e'} \right) \\ &\leq |\{\mathbf{x} + l_e \chi_e\}| \max_{e' \in \{\mathbf{x} + l_e \chi_e\}} f(\chi_{e'}), \end{aligned}$$

which is a contradiction. That means, when an element e arrives, $l_e \chi_e$ may be added to the current set only if $v \leq Km/\alpha$. In addition, as Singleton() returns a vector \mathbf{x} with $f(\mathbf{x}) \geq m$, we don't need to consider the cases whose the parameter v is less than m in DynamicMRT(ε, α). Similar to Theorem 3.1, we receive that this is a $(1/3 - \varepsilon)$ -approximation algorithm.

As the amount of the parameters in the set \mathcal{I} is $O(\log K/\varepsilon)$, and the size of a solution is $O(K)$, the space complexity of the algorithm is $O(K \log K/\varepsilon)$. Meanwhile, for each element and per v , the number of queries for the binary search algorithm is $O(\log K)$. Thus, the per-element query number is $O(\log^2 K/\varepsilon)$.

4 Conclusions

In this paper, we study the problem of maximizing a DR-submodular function under a knapsack constraint on the integer lattice. We design MarginalRatio Thresholding (α, v) algorithm and DynamicMRT(ε, α) algorithm. Here, we introduce a binary search algorithm as a subroutine to decide the level of each kept element. First, we give a one pass off line algorithm combining MarginalRatio

Thresholding (α, v) and the Singleton algorithm with the known OPT. Second, by estimating the OPT, we gain a two passes algorithm. Finally, by updating the estimation interval of the OPT with the arriving element, we receive a one pass streaming algorithm called DynamicMRT (ε, α) . We obtain that it is a $(1/3 - \varepsilon)$ -approximation algorithm with space complexity $O(K \log K/\varepsilon)$ and the query complexity $O(\log^2 K/\varepsilon)$.

Acknowledgements. The first author is supported by Natural Science Foundation of Shandong Province (Nos. ZR2017LA002, ZR2019MA022), and Doctoral research foundation of Weifang University (No. 2017BS02). The second author is supported by National Natural Science Foundation of China (No. 11871081). The fourth author is supported by National Natural Science Foundation of China (No. 12001025) and Science and Technology Program of Beijing Education Commission (No. KM201810005006).

References

1. Badanidiyuru, A., Mirzasoleiman, B., Karbasi, A., Krause, A.: Streaming submodular maximization: massive data summarization on the fly. In: Proceedings of KDD, pp. 671–680 (2014)
2. Buchbinder, N., Feldman, M., Schwartz, R.: Online submodular maximization with preemption. In: Proceedings of SODA, pp. 1202–1216 (2015)
3. Balkanski, E., Rubinfeld, A., Singer, Y.: An exponential speedup in parallel running time for submodular maximization without loss in approximation. In: Proceedings of SODA, pp. 283–302 (2019)
4. Călinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a monotone submodular function subject to a matroid constraint. *SIAM J. Comput.* **40**(6), 1740–1766 (2011)
5. Chakrabarti, A., Kale, S.: Submodular maximization meets streaming: matchings, matroids, and more. *Math. Program.* **154**, 225–247 (2015)
6. Chekuri, C., Quanrud, K.: Submodular function maximization in parallel via the multilinear relaxation. In: Proceedings of SODA, pp. 303–322 (2019)
7. Chekuri, C., Quanrud, K.: Randomize MWU for positive LPs. In: Proceedings of SODA, pp. 358–377 (2018)
8. Das, A., Kempe, D.: Algorithms for subset selection in linear regression. In: Proceedings of STOC, pp. 45–54 (2008)
9. Das, A., Kempe, D.: Submodular meets spectral: greedy algorithms for subset selection, sparse approximation and dictionary selection. In: Proceedings of ICML, pp. 1057–1064 (2011)
10. El-Arini, K., Guestrin, C.: Beyond keyword search: discovering relevant scientific literature. In: Proceedings of ICKDDM, pp. 439–447 (2011)
11. Ene, A., Nguyen, H.L.: Submodular maximization with nearly-optimal approximation and adaptivity in nearly-linear time. In: Proceedings of SODA, pp. 274–282 (2019)
12. Gong, S., Nong, Q., Liu, W., Fang, Q.: Parametric monotone function maximization with matroid constraints. *J. Global Optim.* **75**, 833–849 (2019)
13. Huang, C., Kakimura, N.: Improved streaming algorithms for maximising monotone submodular functions under a knapsack constraint. In: Proceedings of WADS, pp. 438–451 (2019)

14. Jiang, Y.J., Wang, Y.S., Xu, D.C., Yang, R.Q., Zhang, Y.: Streaming algorithm for maximizing a monotone non-submodular function under d-knapsack constraint. *Optim. Lett.* **14**(5), 1235–1248 (2020)
15. Kapralov, M., Post, I., Vondrák, J.: Online submodular welfare maximization: greedy is optimal. In: *Proceedings of SODA*, pp. 1216–1225 (2012)
16. Khanna, R., Elenberg, E.R., Dimakis, A.G., Negahban, S., Ghosh, J.: Scalable greedy feature selection via weak submodularity. In: *Proceedings of ICAIS*, pp. 1560–1568 (2017)
17. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions. *Math. Program.* **14**, 265–294 (1978)
18. Norouzi-Fard, A., Tarnawski, J., Mitrovic, S., Zandieh, A., Mousavifar, A., Svensson, O. Beyond 1/2-approximation for submodular maximization on massive data streams. In: *Proceedings of ICML*, pp. 3829–3838 (2018)
19. Sviridenko, M.: A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.* **32**(1), 41–43 (2004)
20. Shioura, A.: On the Pipage rounding algorithm for submodular function maximization—a view from discrete convex analysis. *Discrete Math. Algorithms Appl.* **1**(1), 1–23 (2009)
21. Soma, T., Kakimura, N., Inaba, K., Kawarabayashi, K.: Optimal budget allocation: theoretical guarantee and efficient algorithm. In: *Proceedings of ICML*, pp. 351–359 (2014)
22. Soma, T., Yoshida, Y.: A generalization of submodular cover via the diminishing return property on the integer lattice. In: *Proceedings of NIPS*, pp. 847–855 (2014)
23. Soma, T., Yoshida, Y.: Maximization monotone submodular functions over the integer lattice. *Math. Program.* **172**, 539–563 (2018)
24. Wolsey, L.: Maximising real-valued submodular set function: primal and dual heuristics for location problems. *Math. Oper. Res.* **7**(3), 410–425 (1982)
25. Wang, Y.J., Xu, D.C., Wang, Y.S., Zhang, D.M.: Non-submodular maximization on massive data streams. *J. Global Optim.* **76**(4), 729–743 (2020)
26. Yu, Q., Xu, E., Cui, S.: Streaming algorithms for news and scientific literature recommendation: submodular maximization with a d-knapsack constraint. In: *Proceedings of IEEE GCSI* (2016)
27. Yang, R.Q., Xu, D.C., Jiang, Y.J., Wang, Y.S., Zhang, D.M.: Approximation robust parameterized submodular function maximization in large-scales. *Asia Pacific J. Oper. Res.* **36**(4), 195–220 (2019)



NASIL: Neural Network Architecture Searching for Incremental Learning in Image Classification

Xianya Fu¹, Wenrui Li¹(✉), Qiurui Chen², Lianyi Zhang², Kai Yang², Duzheng Qing², and Rui Wang¹

¹ Beihang University, Beijing, China
liwenrui@buaa.edu.cn

² Science and Technology on Special System Simulation Laboratory, Beijing Simulation Center, Beijing 100854, China

Abstract. “Catastrophic forgetting” and scalability of tasks are two major challenges of incremental learning. Both of these issues were related to the insufficient capacity of machine learning model and the insufficiently trained weights as the increasing of tasks. In this paper, we try to figure out the impact of the neural network architecture to the performance of incremental learning in the case of image classification. During the increasing of tasks, we propose to use neural network architecture searching (NAS) to find a structure that fits the new tasks collection better. We build a NAS environment with reinforcement learning as the searching strategy and Long Short-Term Memory network as the controller network. Computation operation and connecting previous nodes are selected for each layer in the search phase. For each time a new group of tasks is added, the neural network architecture is searched and reorganized according to the training data set. To speed up the searching, we design a parameter sharing mechanism, in which the same building blocks in each layer share a group of parameters. We also introduce the quantified-parameter building blocks into the NAS, to identify the best candidate during each round of searching. We test our solution in cifar100 data set, the average accuracy outperforms the current representative solutions (LwEMC, iCaRL, GANIL) by 24.92%, 5.62%, and 3.6%, respectively, the more tasks added, the better our solution performs.

Keywords: Continual learning · Network architecture searching · Image classification

Incremental learning is an essential branch of machine learning. It can be used to solve the two major problems in machine learning: 1. the training data set and tasks are incrementally received over time by the machine learning models and the applications need to be applied before all the training data are ready. 2. As typical deep learning training requires all data to be trained and the

amount of training data is too large to be loaded into memory, the data will be repeatedly transferred between disk and memory, which can make the training time unacceptable. Incremental learning is a progressive learning technique [5] that does not require obtaining all training data for all tasks before training. Ideal incremental learning can gradually add new data and learn tasks during the lifetime of machine learning applications while retaining the knowledge of the old tasks.

However, incremental learning faces two problems. The first one is “catastrophic forgetting” [14]. “Catastrophic forgetting” is the knowledge damage of a pre-trained machine learning model during the learning of new tasks. Machine learning models change parameters during the new phrase of learning, but the changing may hurt the previous knowledge. But the real sources of “catastrophic forgetting” is seldom explored. The second critical issue of incremental learning is the scalability of the machine learning model. As the learning continues, the learning model needs to be scalable; one fixed-size model can not perform both well at the very beginning when the data and tasks are very few and at the end when the data and tasks are quite large.

Previous studies on incremental learning have mostly focused on the weights of neural networks. However, when we look at the impact of weights and network architecture on the effect of incremental tasks separately, we believe that the correspondence between neural network architecture and incremental tasks is important. In [8], the neural network architecture can make the model have higher potential than the weight. The performance of a few network architecture is better than other models for some specific tasks.

The main contribution of this work is to use neural network architecture search technology to search for the most suitable network model based on the original model architecture when new tasks are added, to improve the adaptability and expansibility of the model in incremental learning. In order to balance the demand for incremental training in memory resources and computing resources, we introduce a quantified cell in search space and use shared weights for search strategy.

The main contributions of this paper are as follows:

- To incremental learning, neural network architecture has a more significant impact on the performance of incremental tasks than weights.
- Propose an incremental NAS to solve the problem of feature set expansion and model adaptability in incremental learning.
- Using quantified cell and shared weights method to balance the gap between memory, computational and incremental learning resource limits.

We find that the quantified NAS obtains higher accuracy than the non-quantized NAS in the same search epoch through experiments, which provides a better basis for the incremental learning training process. In the final experiment, our method is obviously superior to [13, 19, 23]. The searched network structure is more suitable for incremental processes.

1 Related Works

Incremental Learning: Traditional approach [5, 15, 17] for incremental learning realized by leveraging linear classifier. Current research on incremental learning focuses on solving the problem of “catastrophic forgetting” and tries to use the fewest samples of the old class. These methods are mainly divided into two types. One type of method limits the variation of network parameters so that it does not significantly deviate from the parameters learned in previous tasks. iCaRL [19] defines class-incremental learning and uses distillation loss to maintain old class information. [23] uses Generative Adversarial Networks to train an old class sample generator so that it does not use the original samples. Another type of method stores the old structure and parameters to remember the previous tasks. [24] uses a well-designed method of reinforcement learning RCL to solve this problem. However, in RCL, the time stamp is used as an activation point for each new data, which limits the flexibility of the model. [21] solves the problem of object detection by storing the old task model to help generate the new model. The rest of the research, for example [20], proposes a Meta-Experience Replay algorithm for non-stationary distribution of data, combining experience replay with optimization based meta-learning for continual learning. [4] generates a sample sorted list based on the distance to the average sample of the class, and filters the most representative samples from the old class samples. However, most studies have not considered model capacity or network architecture. Our research focuses on using an incremental NAS to filter the architecture of each new task.

Neural Architecture Search: NAS consists of three parts: search space, search strategy, and performance estimation strategy. In search space, main structures include chain structure [2, 6], multi-branch structure [3, 25] and dubbed cell/block structure [10]. In [12], a novel search space pruning method was used to construct a network architecture search space by Partial Order Assumption to automatically search for architecture with the best latency and accuracy trade-off. Most of the research is focused on the search strategy. It mainly includes the following: evolutionary approaches [18], reinforcement learning [1, 25], Bayesian optimization, and other learning algorithms. In order to adapt to different hardware platforms, some research has emerged for the performance estimation strategy. These studies not only consider valid accuracy as the only indicator, but also consider actual latency [22], model size [7, 9] and so on. This paper attempts to add a quantization algorithm to the NAS to reduce the model size with almost no loss of model accuracy. More importantly, under the mechanism of shared NAS, the weight parameter using fewer bits will result in higher accuracy in the same number of iterations.

2 Methodology

In this section, we will formulate the incremental learning problem and propose our new idea. We believe that the impact of network architecture on incremental

learning tasks is important. As tasks increase, simply increasing model capacity (such as the number of layers) may not work. Therefore, our basic idea is to use neural network architecture search algorithm to select an architecture that is more suitable for the current task set. Finally, it is compared with the current state-of-the-art incremental learning algorithm in the Experiment.

Incremental learning means that a learning system can continuously learn knowledge from new samples and preserve most of the knowledge that it has learned before.

For a sequential data stream, we define the data transmitted in the i time as d_i . d_i can be divided into two parts, one for training and the other one for testing. We consider the validation set as part of the test set and are not specifically pointed out below.

$$d_i = d_{T_i} \cup d_{V_i}$$

where d_{T_i} represents the data used for training in the incoming data, and d_{V_i} represents the data used for testing. We define c_i as the incoming data categories, D_{T_i} as the training set for incremental learning, and D_{V_i} as the testing set for incremental learning.

D_{T_i} is calculated as follow:

$$D_{T_i} = \begin{cases} d_{T_0}, i = 0 \\ d_{T_i} \cup d_{T_{i-1}}(\frac{\theta}{c_{i-1}}) \cup d_{T_{i-2}}(\frac{\theta}{c_{i-2}}) \dots \cup d_{T_0}(\frac{\theta}{c_0}), i > 0 \end{cases}$$

D_{V_i} as calculated as follow:

$$D_{V_i} = \begin{cases} d_{V_0}, i = 0 \\ d_{V_i} \cup d_{V_{i-1}} \cup d_{V_{i-2}} \dots \cup d_{V_0}, i > 0 \end{cases}$$

θ represents the total number of old samples that will be maintained throughout the incremental learning process. Controlling the total number of old class samples to a fixed value helps control the size of the training set, thereby controlling memory usage. This approach will avoid the explosive growth of memory usage during incremental learning.

We use M_i to represent the incremental learning model obtained at the i moment. M_i will be trained by D_{T_i} and tested at D_{V_i} . The goal of incremental learning is to learn the best accuracy for each time.

$$\text{maxAccuracy}(M_i, D_{V_i})$$

The general incremental learning algorithm focuses on limiting network parameters or using the method of reinforcement learning to retain the old knowledge, but only paying attention to this part can not guarantee the long-term stability of incremental learning. As the number of data categories increases, a fixed-architecture network limits the ability of the classifier to maintain long-term effectiveness. Therefore, we focus on network capacity. However, we are not sure whether there is a linear relationship between the data categories and

network architecture. It is not the best way to directly increase the network architecture as the data categories increases. While retaining the ancient knowledge, we consider using NAS to search out the network architecture suitable for the current data, to more effectively enhance the classification ability of multi-class classifiers. On this basis, the searched network architecture is further trained to obtain the final multi-classifier. The overall process is shown in Fig. 1.

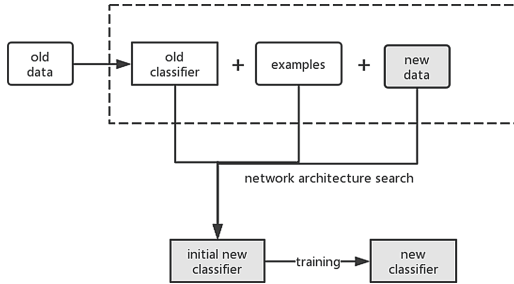


Fig. 1. NAS incremental learning process, where examples are partial old class samples. In the search phase, the network architecture search is performed based on the old class classifier, some old class samples, and all-new class samples, thereby obtaining a network structure for the current category data. In the training phase, the training is continued based on the network structure of the search phase.

3 Neural Network Architecture Search for Incremental Learning

For incremental tasks, NAS reconstructed the model architecture and selected new model capacity. However, there are still two problems in this process: 1) The search time is too long, which dramatically increases the time required to obtain the classifier. 2) Excessive memory consumption, which is contrary to the way incremental learning selects fewer old class samples in order to use less memory. We use the sharing of weights and quantizing cells to solve these two problems. Next, we will discuss two parts: one is about incremental neural network architecture search and the other is about shared weights and quantifies method.

3.1 Incremental Neural Network Architecture Search

In this section, we introduce a neural network architecture search approach for incremental learning. Incremental NAS uses both shared parameters and knowledge distillation. Knowledge distillation helps preserve ancient knowledge so that the impact of old classes can be considered in the process of searching the network architecture. Sharing parameters will shorten the training time during the

searching and after the increasing of the new category. This approach is similar to using a pre-trained model classes are added so that incremental learning can maintain long-term stability.

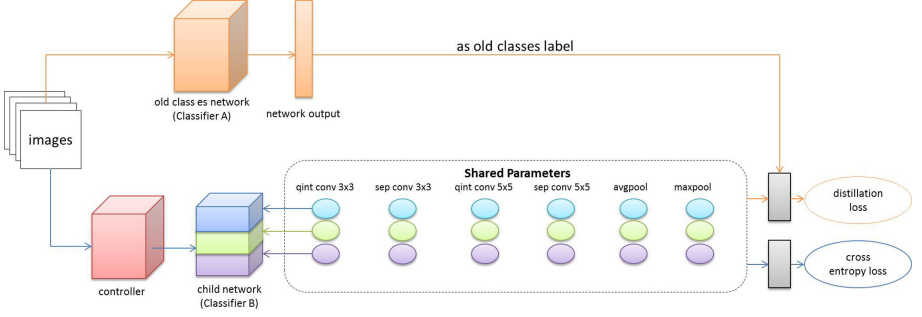


Fig. 2. Shared parameters quantified incremental NAS process

Unlike the other NAS methods, the shared parameters NAS maintains a child network weights set. Each time the child network is built, the controller generates the network architecture and then takes the weight parameters from the set before training. The training in the NAS does not train the child network to converge fully, but only simple training, such as several forward and backward propagation for the current entire data set. At the same time, the shared parameters set is continuously updated according to the weights of the child model.

When the first data stream comes, we use the data as D_{T_0} to filter out the network architecture that best fits the current data set. In the first search process, the shared weight parameters set is initialized. According to the child network selected by the controller, the weights are taken from the shared weight parameters set, and the process is looped to train the final model M_1 of the first data stream.

When the second data stream comes, the data set D_{T_2} is generated, and the multi-class classification model is trained according to Fig. 2. Based on the last NAS process, we search for new child network. The loss in the search process consists of two parts, cross entropy loss, and distillation loss.

Take the second data stream as an example. The model prediction result p_x is:

$$p_x = \text{sigmoid}(\text{logits}_x) = \frac{1}{1 + e^{-\text{logits}_x}}$$

Cross entropy loss is written as:

$$L_c = \frac{1}{n_1} \sum_{(x,y) \in D_{T_2} - D_{T_1}} \sum_{i=1}^{n_1} -[y_i \times \ln p_{x_i} + (1 - y_i) \times \ln(1 - p_{x_i})]$$

n_1 is the number of elements in $D_{T_2} - D_{T_1}$. x is the input and y is the one-hot label. logists_x is the model prediction result. Distilling loss is formulated as follows:

$$L_d = \frac{1}{n_2} \sum_{(x) \in D_{T_2} \cap D_{T_1}} \sum_{i=1}^{n_2} -[y_i' \times \ln p_{x_i} + (1 - y_i' \times \ln(1 - p_{x_i}))]$$

n_2 is the number of elements in $D_{T_2} \cap D_{T_1}$, and y' is the prediction result after running the data belonging to the old category in the new data set D_{T_2} using M_1 . Then the overall loss is:

$$L = \lambda L_d + (1 - \lambda) L_c$$

The loss value is used to train the model M_2 at the second point in time during the quantified NAS search phase and the training phase. The subsequent incremental learning process can be followed by this analogy.

Based on this, we proposed a quantified NAS. We convert the full-precision value in the search space into 8-bit fixed-point value. More importantly, we find the reduction in the number of data bits allows for greater accuracy in the same epoch of the NAS.

4 Experiments

In this part, we first experimented with the incremental learning process using the cifar100 [11] dataset, adding 5 or 10 categories at a time and comparing it with the other five methods. Secondly, the cifar10 data set is used to compare the NAS with and without quantized nodes.

We used deep learning framework tensorflow and one Tesla V100 GPU. In the following experimental sections, all experiments involving NAS, whether quantified or not. The quantified NAS is 3×3 quantized convolution, 5×5 quantized convolution, 3×3 deep convolution, 5×5 deep convolution, average pooling, and maximum pooling. Non-quantified NASs are 3×3 32-bit convolution, 5×5 32-bit convolution, 3×3 deep convolution, 5×5 deep convolution, average pooling, and maximum pooling.

4.1 Overall Performance of NASIL

We use the quantified NAS approach to screen out the network architecture with the highest valid accuracy in 100 epochs. Continue training for the current optimal network structure to obtain test accuracy. In the search phase, we set the number of layers to 18, the out filter size of first layer to 16, and the remaining parameters are the same as [16]. We performed two experiments, increasing five categories and increasing ten categories each time, and recorded the highest valid accuracy during the network architecture search.

We compare our results with: iCaRL [19], LwEMC [13], GANIL [23], fixed representation and finetune. These five methods all use Resnet32, and the parameter settings are the same as [19].

Table 1. Average test accuracy for each increment of 10 classes and 5 classes for each increment

	NASIL	GANIL	iCaRL	LwEMC	Fixed representation	Finetuning
5 classes	66.00%	63.85%	60.86%	33.05%	17.93%	16.93%
10 classes	69.62%	66.02%	64.00%	44.70%	28.10%	26.10%

- fixed representation(fixed repre): After the first training, the feature representation parameters is fixed during each increasing time.
- finetuning: Training based on new samples, no restrictions on parameters.

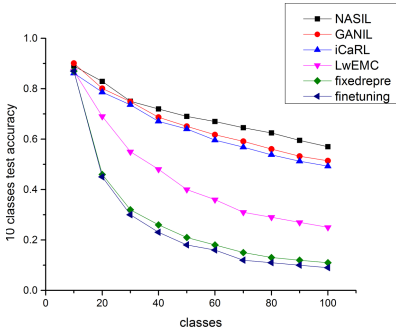


Fig. 3. Multi-class accuracy on iCIFAR-100 with 10 classes per batch

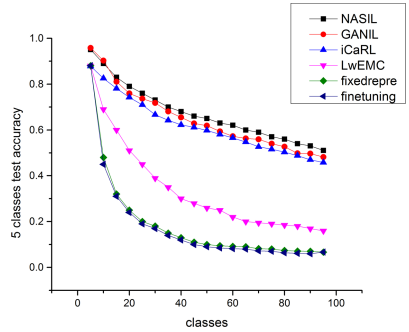


Fig. 4. Multi-class accuracy on iCIFAR-100 with 5 classes per batch

As can be seen from Fig. 4 and Fig. 3, our method has apparent advantages in accuracy. Compared to iCaRL [19], which directly uses knowledge distillation to preserve ancient knowledge. At the same time, our approach is more stable in the later stages of incremental learning than using GAN [23] to generate the same number of old class samples.

We averaged the 20 test accuracy obtained in each of the five new classes and averaged the 10 test accuracy obtained in each of the ten new classes. As shown in Table 1, we can intuitively see that the use of dynamic network architecture can make incremental learning more stable and have better scalability for new tasks.

4.2 Searched Network Architecture

The number of network weight parameters can be seen as a measure of network capacity. In order to further discover the relationship between network capacity and data categories, we studied the network structures and the number of weight parameters that were filtered in each increasing time. We compared the node operations that the NAS screened out in the 10-class incremental process. The results are shown in Fig. 5. It should be noted that we do not show the connection between layers in the figure.

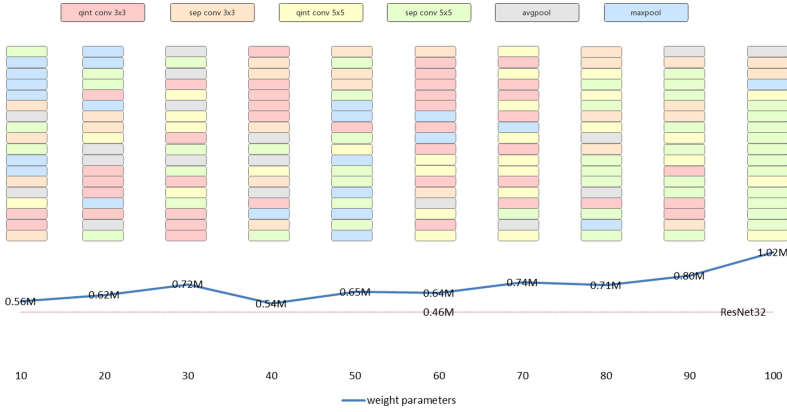


Fig. 5. NAS search architecture comparison. The upper part is the 10 models that are searched out during each incremental learning process of 10 categories. The lower part is the curve of the number of parameters in each network architecture.

In the process of increasing the number of categories, the number of network weight parameters are increasing. Therefore, it can also be explained that as the number of categories increases, the network architecture search tends to select a structure with larger network capacity. At the same time, in the less-category phase, NAS tends to choose a network with more pool operations, and when the category is increased to a certain stage, NAS prefers to choose convolution operations.

The increase in network capacity may be related to data set indicators such as the number of data categories, the number of samples per class, the similarity between classes and classes. It is difficult to directly determine the relationship between network capacity and data sets through functional methods. The neural network architecture search method provides a dynamic network structure for the incremental learning process, which makes the network more capable of classifying new tasks.

4.3 Quantified NAS

In this part of the experiment, we experiment on quantifying NAS to verify it’s effectiveness. Firstly, for cifar10 [11], two network structures are filtered out by using quantified convolution operation and full-precision convolution operation respectively and we compare their accuracy, inference time and model size on Huawei mate10. Secondly, we filter the network architecture for the cifar100 with incremental learning process and obtain the highest valid accuracy for comparison.

Table 2. Test accuracy, inference time, and model size obtained during NAS using quantized convolutional layers and full-precision convolutional layers

	float32	qint8
Accuracy	92.82%	92.59%
Inference time (ms)	885	328
Model size (Mb)	6.2	2.8

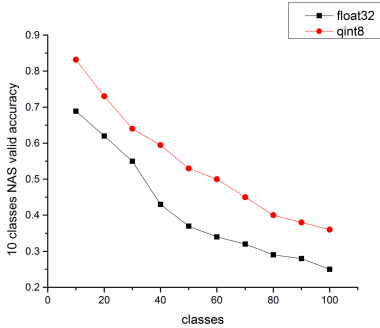


Fig. 6. The highest valid accuracy in the NAS process using float32 and qint8 each time increasing 5 classes

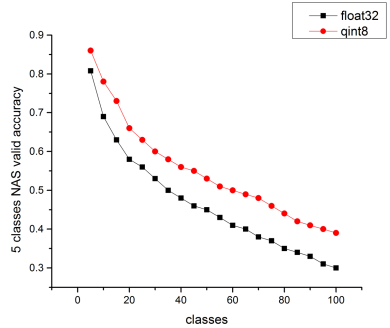


Fig. 7. The highest valid accuracy in the NAS process using float32 and qint8 each time increasing 10 classes

On cifar10 dataset, we set the number of layers to 12, and performed 310 iterations. This process does not involve incremental learning. We compared data using 32-bit convolution operations and 8-bit quantized convolution operations in the search space. We found that the NAS process with 8-bit quantization operation can achieve higher test accuracy than the full-precision operation in the same epoch. The main reason for making 8-bit quantized convolution operations more accurate is that weight sharing is used in the process of network

architecture search, while 32-bit data is more “accurate” than 8-bit data. This will allow 32-bit data to be rigorously adjusted to achieve higher accuracy.

We tested the accuracy, inference time, and model size of the two models on Huawei mate10. The results are shown in Table 2. The accuracy of the quantized model is slightly lower than that of the unquantized model, but it is almost negligible. However, the inference time becomes shorter, and the model size becomes smaller. Inaccurate calculations take some loss of precision, which brings advantages of time and memory.

At the same time, we compared the difference in the valid accuracy obtained by using the quantified NAS and directly using the NAS when incrementally learning cifar100. The results can be shown in Fig. 6 and Fig. 7. The NAS process that joins the qint8 convolution node can achieve higher valid accuracy.

5 Conclusion

Incremental learning or lifelong learning is a persistent problem in the field of machine learning. Compared with weights, network architecture has a more significant impact on the performance of incremental tasks. We proposed the NASIL approach to use shared weights and quantified NAS to solve some of the problems of catastrophic forgetting and model scalability. The accuracy of incremental learning is improved by nonlinearly increasing the network capacity. We show the experimental results on cifar10 and cifar100, and our approach has advantages in accuracy.

In this paper, we provide a solution for using dynamic models in incremental learning. When new NAS technologies emerge, we can still use the proposed method to filter the models for incremental learning. Similarly, the quantified structure can also be used in the search space of the remaining NAS.

Acknowledgement. This work is supported by the National Key Research and Development Program of China under grant 2018YFB0203901. This work is also supported by the NSF of China under grant 61732002.

References

1. Baker, B., Gupta, O., Naik, N., Raskar, R.: Designing neural network architectures using reinforcement learning. In: International Conference on Learning Representations (2017)
2. Cai, H., Chen, T., Zhang, W., Yu, Y., Wang, J.: Reinforcement learning for architecture search by network transformation. CoRR abs/1707.04873 (2017). <http://arxiv.org/abs/1707.04873>
3. Cai, H., Yang, J., Zhang, W., Han, S., Yu, Y.: Path-level network transformation for efficient architecture search. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 678–687. PMLR, Stockholmsmässan, Stockholm Sweden, 10–15 July 2018. <http://proceedings.mlr.press/v80/cai18a.html>

4. Castro, F.M., Marín-Jiménez, M.J., Guil, N., Schmid, C., Alahari, K.: End-to-end incremental learning. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11216, pp. 241–257. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01258-8_15
5. Cauwenberghs, G., Poggio, T.: Incremental and decremental support vector machine learning. In: Advances in Neural Information Processing Systems, pp. 409–415 (2001)
6. Chollet, F.: Xception: deep learning with depthwise separable convolutions. CoRR abs/1610.02357 (2016). <http://arxiv.org/abs/1610.02357>
7. Dong, J.-D., Cheng, A.-C., Juan, D.-C., Wei, W., Sun, M.: DPP-Net: device-aware progressive search for pareto-optimal neural architectures. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11215, pp. 540–555. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01252-6_32
8. Gaier, A., Ha, D.: Weight agnostic neural networks. CoRR abs/1906.04358 (2019). <http://arxiv.org/abs/1906.04358>
9. Hsu, C.H., et al.: MONAS: multi-objective neural architecture search using reinforcement learning. arXiv preprint [arXiv:1806.10332](https://arxiv.org/abs/1806.10332) (2018)
10. Huang, G., Liu, Z., Der Maaten, L.V., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition (2016)
11. Krizhevsky, A.: Learning multiple layers of features from tiny images (2009)
12. Li, X., Zhou, Y., Pan, Z., Feng, J.: Partial order pruning: for best speed/accuracy trade-off in neural architecture search. In: Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition (2019)
13. Li, Z., Hoiem, D.: Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(12), 2935–2947 (2017)
14. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: the sequential learning problem. *Psychol. Learn. Motiv.* **24**, 109–165 (1989)
15. Mensink, T., Verbeek, J., Perronnin, F., Csurka, G.: Distance-based image classification: generalizing to new classes at near-zero cost. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(11), 2624–2637 (2013)
16. Pham, H., Guan, M.Y., Zoph, B., Le, Q.V., Dean, J.: Efficient neural architecture search via parameter sharing. arXiv preprint [arXiv:1802.03268](https://arxiv.org/abs/1802.03268) (2018)
17. Polikar, R., Udpa, L., Udpa, S., Honavar, V.: An incremental learning algorithm for supervised neural networks. *IEEE Trans. SMC (C), Special Issue on Knowledge Management* (2000)
18. Real, E., et al.: Large-scale evolution of image classifiers. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70, pp. 2902–2911. JMLR. org (2017)
19. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: iCaRL: incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2001–2010 (2017)
20. Riemer, M., et al.: Learning to learn without forgetting by maximizing transfer and minimizing interference. arXiv preprint [arXiv:1810.11910](https://arxiv.org/abs/1810.11910) (2018)
21. Shmelkov, K., Schmid, C., Alahari, K.: Incremental learning of object detectors without catastrophic forgetting. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3400–3409 (2017)
22. Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., Le, Q.V.: MNASNet: platform-aware neural architecture search for mobile. In: Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, pp. 2820–2828 (2019)

23. Wu, Y., et al.: Incremental classifier learning with generative adversarial networks. arXiv preprint [arXiv:1802.00853](https://arxiv.org/abs/1802.00853) (2018)
24. Xu, J., Zhu, Z.: Reinforced continual learning. In: Advances in Neural Information Processing Systems, pp. 899–908 (2018)
25. Zoph, B., Vasudevan, V., Shlens, J., Le, Q.V.: Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition (2017)



Nonsubmodular Maximization with Knapsack Constraint via Multilinear Extension

Jiachen Ju, Min Li, Jianxin Liu, Qian Liu^(✉), and Yang Zhou

School of Mathematics and Statistics, Shandong Normal University,
Jinan 250014, People's Republic of China
jiachen8201@163.com, liminEmily@sdsu.edu.cn, ljx250031@163.com,
lq-qsh@163.com, zhyg1212@163.com

Abstract. For the problem of maximizing a monotone submodular function subject to knapsack constraint, there is a $(1 - 1/e - \epsilon)$ -approximation algorithm running a nearly-linear time. In this paper, we consider the case that the objective function is nonsubmodular. We propose an approximation algorithm with approximation ratio

$$\kappa \left(1 + \frac{\epsilon}{\kappa^2(1-\epsilon)} \right)^{-1} \left(1 - e^{-\Omega(\epsilon^2/\lambda)} \right) \left(1 - e^{-\kappa^3} - O(\epsilon) \right),$$

and complexity $\tilde{O}\left(\frac{1}{1-\tau}n^2(\log n)^{\frac{1}{\epsilon}+2}\right)$, where κ is the continuous submodularity ratio, τ is the curvature and λ is the largest weight. The technology of our algorithm is using continuous greedy to get a fractional solution and then rounding it with the contention resolution scheme.

Keywords: Submodularity ratio · Knapsack constraint · Continuous greedy · Contention resolution scheme

1 Introduction

1.1 Background

In this paper, we consider the problem of maximizing a monotone function under a knapsack constraint, which has a wide range of applications in selection of investments and portfolios [9], selection of assets for asset-backed securitization [10], generating keys for the Merkle-Hellman [11], and other knapsack cryptosystems. Especially, given a ground set E , a monotone, nonnegative and normalized function $f : 2^E \rightarrow \mathbb{R}_+$ with submodularity ratio and a weight function $w : E \rightarrow [0, 1]$, the goal is to deal with the following optimization problem:

$$\max\{f(S) : \sum_{e \in S} w(e) \leq 1, S \subseteq E\}.$$

Supported by National Natural Science Foundation of China (No. 12001335) and Shandong Provincial Natural Science Foundation, China (Nos. ZR2020MA029, ZR2019PA004).

There are lots of works for the knapsack constraint problem when the function f is submodular. In [12], Sviridenko proposes a $(1 - e^{-1})$ -approximation algorithm by employing the ideas of density greedy, in which the complexity of function value oracle is $O(n^5)$. Badanidiyuru and Vondrák [1] give a variant of the continuous greedy algorithm with $(1 - e^{-1} - \epsilon)$ -approximation and $O(n^2(\log n/\epsilon)^{1/\epsilon^8})$ oracle complexity. Moreover, Ene and Nguyễn [5] improve the oracle complexity by presenting a nearly-linear time algorithm with the near-optimal approximation ratio. Yoshida [14] considers maximizing a monotone submodular function with curvature τ , and acquires a better $(1 - \tau/e - \epsilon)$ -approximation ratio.

There are also some results for the case that f is nonsubmodular with different constraints. Some kinds of submodularity ratio are introduced to measure the proximity to submodularity. For the cardinality constraint case, Bian et al. [2] exert the greedy algorithm to produce a tight approximation guarantee of $\frac{1}{\alpha}(1 - e^{-\alpha\beta})$ where β and α are the greedy submodularity ratio and greedy curvature. In [16], Zhang et al. extend the results to the knapsack constraint case. The oracle complexity of the algorithm is dependent on the submodularity ratio and curvature, which is at least $O(n^5)$ without the curvature information. For the matroid constraint case, Gong et al. [8] propose the submodularity ratio γ and provide a $\gamma(1 - e^{-1})(1 - e^{-\gamma} - o(1))$ -approximation algorithm.

1.2 Our Techniques

In this paper, we consider the nonsubmodular knapsack constrained maximization problem with a lower complexity. Our strategy contains two steps. Firstly, we maximize the continuous function $F(\mathbf{x})$, the multilinear extension of the function f , and obtain a fractional solution. Secondly, we round it to a feasible solution.

The continuous setting of our algorithm framework resembles those continuous treatment in [14]. First, we divide E into big elements set and small elements set according to the values of its elements. We assume that there are m big elements in O , and denote $O_B = \{o_1, o_2, \dots, o_m\}$, $O_S = O \setminus O_B$. In each step of the continuous greedy process, we find those elements $e_i \in E$ such that (a) $F(\mathbf{x} \vee \mathbf{1}_{e_i}) \geq F(\mathbf{x} \vee \mathbf{1}_{o_i})$, (b) $w(e_i) \leq w(o_i)$. Afterwards, we update current vector \mathbf{x} by adding $\epsilon \mathbf{1}_{e_i}$. For small elements, we find a vector \mathbf{v} such that (a) the marginal value of \mathbf{v} is at least that of O_S , (b) the weight of \mathbf{v} is at most that of O_S . Next, the current \mathbf{x} is added with $\epsilon \mathbf{v}$.

A challenging difficulty is that the multilinear extension $F(\mathbf{x})$ is no longer concave along any line of direction $\mathbf{d} \geq 0$ when the function f is nonsubmodular. To overcome this difficulty, we adopt the contention resolution (CR) scheme to round the fractional solution. The CR scheme can be traced back to [4], which provides a useful and general framework with respect to the knapsack polytope.

To summarize, we extend the problem of maximizing submodular function under a knapsack constraint to the case of nonsubmodular function, and present

a polynomial-time algorithm with a gratifying approximation ratio as

$$\kappa \left(\frac{1}{1 + \frac{\epsilon}{\kappa^2(1-\epsilon)}} \right) \left(1 - e^{-\Omega(\epsilon^2/\lambda)} \right) \left(1 - e^{-\kappa^3} - O(\epsilon) \right),$$

and the oracle complexity as $\tilde{O}(\frac{1}{1-\tau} n^2 (\log n)^{\frac{1}{\epsilon}+2})$, where κ is the submodularity ratio, τ is the curvature and λ is the largest weight.

The paper is organized as follows. In Sect. 2, we introduce some foundational definitions and related lemmas. In Sect. 3, we propose a continuous greedy algorithm. In Sect. 4, we apply the CR scheme to round the fractional solution and show that the loss is little. In Sect. 5, we present the complete algorithm and analyze the time complexity.

2 Preliminaries

2.1 Notations

We denote the ground set by $E = \{1, 2, \dots, n\}$. The function $f : 2^E \rightarrow \mathbb{R}_+$ is said to be monotone (nondecreasing) if $f(S) \leq f(T)$ for all $S \subseteq T \subseteq E$. Moreover, f is said to be nonnegative and normalized when $f(\cdot) \geq 0$ and $f(\emptyset) = 0$. For the weight function $w : E \rightarrow [0, 1]$ and a subset $S \subseteq E$, we denote $w(S) = \sum_{e \in S} w(e)$. Given a vector $\mathbf{x} \in [0, 1]^n$ and a set $S \subseteq E$, we denote $\mathbf{x}(S) = \sum_{e \in S} \mathbf{x}(e)$. Given an element $e \in E$, the vector $\mathbf{1}_e$ means that the e -th coordinate is 1, while the others are 0. Moreover, for a subset $S \subseteq E$, we define $\mathbf{1}_S = \sum_{e \in S} \mathbf{1}_e$.

Given an integer $n \in \mathbb{N}$, let $[n]$ be the set $\{1, \dots, n\}$ for compactness. Also, for a function f and an element $e \in E$, we use $f(e)$ instead of $f(\{e\})$. For simplicity, we denote $f_S(T) = f(S \cup T) - f(S)$ as the marginal gain of set T to S . Further, The function f is called submodular if, for any subset $S \subseteq T$ and any $e \in E \setminus T$, $f_S(e) \geq f_T(e)$. If $f(\cdot)$ is submodular, the function $f_S(\cdot)$ is also submodular obviously.

2.2 The Multilinear Extension

The main tool we will use in this paper is the multilinear extension, which is a continuous relaxation of objective function f . Given a vector $\mathbf{x} \in [0, 1]^n$, we define $R(\mathbf{x})$ as a random set such that each element is chosen independently with probability $\mathbf{x}(e)$.

Definition 2.1 (Multilinear extension, [3]). For a monotone function $f : 2^E \rightarrow \mathbb{R}$, the multilinear extension $F : [0, 1]^n \rightarrow \mathbb{R}$ of f is defined as

$$F(\mathbf{x}) := \mathbf{E}[f(R(\mathbf{x}))] = \sum_{S \subseteq E} f(S) \prod_{e \in S} \mathbf{x}(e) \prod_{e \in E \setminus S} (1 - \mathbf{x}(e)).$$

Proposition 2.1 (Property of multilinear extension, [6,8]). *For an element $e \in E$ and a vector $\mathbf{x} \in [0, 1]^n$, we define $\partial_e F(\mathbf{x})$ as the slope of F at \mathbf{x} in the direction of $\mathbf{1}_e$. The following fact is well known:*

$$\partial_e F(\mathbf{x}) = \frac{\mathbf{E} [f_{R(\mathbf{x})}(e)]}{1 - x_e} = \mathbf{E}_{R(\mathbf{x}) \subseteq E \setminus e} [f_{R(\mathbf{x})}(e)].$$

2.3 Submodularity Ratio and Curvature

For nonsubmodular functions, the submodularity ratio is introduced in [8] to describe the proximity to submodularity.

Definition 2.2 (Generic submodularity ratio, [8]). *Given a ground set E and a monotone set function $f : 2^E \rightarrow \mathbb{R}_+$, the generic submodularity ratio of f is the largest scalar κ' such that for any $S \subseteq T \subseteq E$ and any $e \in E \setminus T$,*

$$f_S(e) \geq \kappa' \cdot f_T(e).$$

For convenience, we call a function f a κ' -submodular function when its submodularity ratio is κ' .

Proposition 2.2 (Property of generic modularity ratio, [8]). *For a monotone set function $f : 2^E \rightarrow \mathbb{R}_+$ with generic submodularity ratio κ' , it holds that*

1. $\kappa' \in (0, 1]$,
2. $f(\cdot)$ is submodular iff $\kappa' = 1$,
3. $\sum_{e \in \Omega \setminus S} f_S(e) \geq \kappa' \cdot f_S(\Omega)$, for any subset $S, \Omega \subseteq E$.

Remark 2.1. A monotone set function $f : 2^E \rightarrow \mathbb{R}_+$ with generic submodularity ratio κ' satisfies generic subadditive property, i.e., for all $\Omega \subseteq E$,

$$f(\Omega) \leq \frac{1}{\kappa'} \sum_{e \in \Omega} f(e).$$

Proposition 2.3. *For a monotone set function $f : 2^E \rightarrow \mathbb{R}_+$ with generic submodularity ratio κ' , it holds that for any subset $A, B \subseteq E$,*

$$\kappa' f(A \cup B) - \kappa' f(B) \leq f(A) - f(A \cap B).$$

Remark 2.2. According to Proposition 2.3, it follows that for any subset $A, B \subseteq E$,

$$\kappa' f(A \cup B) \leq f(A) + f(B).$$

Proposition 2.4. *For a monotone set function $f : 2^E \rightarrow \mathbb{R}_+$ with generic submodularity ratio κ' , it follows that for every subset $S \subseteq T \subseteq E$ and $X \subseteq E$,*

$$f_S(X) \geq \kappa' \cdot f_T(X).$$

Proof. One can set $A = S \cup X$, $B = S \cup (T \setminus X)$. Notice that $A \cap B = S$, $A \cup B = T$. And it follows Proposition 2.3 and the monotonicity that

$$f(S \cup X) - f(S) \geq \kappa' f(T \cup X) - \kappa' f(S \cup (T \setminus X)) \geq \kappa' f(T \cup X) - \kappa' f(T).$$

The definition of submodularity ratio can also be extended to the multilinear extension.

Definition 2.3 (Continuous generic submodularity ratio, [15]). Given a monotone function $f : 2^E \rightarrow \mathbb{R}$, the continuous generic submodularity ratio of f is defined as the largest κ such that

$$\partial_e F(\mathbf{x}) \geq \kappa \partial_e F(\mathbf{y}),$$

for any $\mathbf{x} \leq \mathbf{y}$ and $e \in E$.

It is obvious to see that $\kappa \leq \kappa'$ by Definition 2.2 and Definition 2.3.

Definition 2.4 (Generalized curvature, [13]). Let f be an arbitrary monotone function. The curvature τ of f is defined as

$$\tau = 1 - \min_{j \in E} \min_{S, T \subseteq E \setminus j} \frac{f_S(j)}{f_T(j)}.$$

3 Solving the Multilinear Relaxation

In this part, we will design an algorithm to produce a fractional solution that could almost reach $(1 - e^{-\kappa'/3})$ -approximation. Define the function $W : [0, 1]^E \rightarrow \mathbb{R}_+$ as:

$$W(\mathbf{x}) = \sum_{e \in E} \mathbf{x}(e)w(e).$$

Let O be the optimal solution in reality and d be the largest singleton function value. Then we have $d \leq f(O) \leq \frac{1}{\kappa'} nd$. Given any $\epsilon \in (0, 1]$, we define $V_{\epsilon, n} = \{nd, (1 - \epsilon)nd, \dots, \epsilon d, 0\}$, such that there exists a $v \in V_{\epsilon, n} / \kappa' \triangleq \{v / \kappa' | v \in V_{\epsilon, n}\}$ satisfying

$$(1 - \epsilon)v - \epsilon d \leq f(O) \leq v.$$

Note that we have $|V_{\epsilon, n}| = O\left(\log_{1/(1-\epsilon)}(n/\epsilon)\right) = O(\log(n/\epsilon)/\epsilon)$. So far the ground set E can be divided into small elements set and big elements set with respect to ϵ and v . An element e is called “small” if

$$f(e) \leq \epsilon^6 v,$$

and “big” otherwise. Let $E_B \triangleq E_B(v) \subseteq E$ be the collection of big elements and $E_S \triangleq E_S(v) = E \setminus E_B(v)$. For the optimal solution O , define $O_S \triangleq O_S(v) = E_S(v) \cap O$ and $O_B \triangleq O_B(v) = E_B(v) \cap O$.

Lemma 3.1. Suppose that the guessed value v follows $(1 - \epsilon)v - \epsilon d \leq f(O) \leq v$, then it holds that $|O_B| \leq \frac{1}{(1-\tau)\epsilon^6}$, where τ is the generalized curvature.

3.1 Arrange Small Elements

In this section, we propose Algorithm 1 to handle the small elements. To update the current solution \mathbf{x} , we find a vector \mathbf{v} by solving a linear programming to satisfy $\sum_{e \in E_S} \mathbf{v}(e) \mathbf{E} [f_{R(\mathbf{x})}(e)] \geq (1-\epsilon)^3 \kappa' \mathbf{E} [f_{R(\mathbf{x})}(O_S)] - \frac{3\epsilon d}{\kappa'}$ and $W(\mathbf{v}) \leq w(O_S)$. For each $e \in E_S$, neither $\mathbf{E} [f_{R(\mathbf{x})}(e)]$ nor $\mathbf{E} [f_{R(\mathbf{x})}(O_S)]$ could be computed in polynomial time. Their values can be estimated by the algorithm ESTIMATE. For the set E_S , we fancy the accuracy of each element to reach $1 - \delta$ at the same time. This can also be well guaranteed, as long as the correct probability of each element is more than $1 - \delta/n$.

The following lemma shows that we can obtain a good approximation of $\mathbf{E} [f_{R(\mathbf{x})}(e)]$ with probability $1 - \delta/n$.

Lemma 3.2. *With probability at least $1 - \delta/n$, we have*

$$(1 - \epsilon) \mathbf{E} [f_{R(\mathbf{x})}(e)] - \frac{\epsilon d}{n\kappa'} \leq \boldsymbol{\theta}(e) \leq (1 + \epsilon) \mathbf{E} [f_{R(\mathbf{x})}(e)] + \frac{\epsilon d}{n\kappa'},$$

for every $e \in E_S$.

Algorithm 1. SMALL ELEMENTS $_{\epsilon, \delta}(f, w, E_S, \gamma, \mathbf{x})$

Input: A monotone κ' -submodular function $f : 2^E \rightarrow \mathbb{R}_+$, a weight function $w : E \rightarrow [0, 1]$, $\epsilon, \delta \in (0, 1)$, a small elements set E_S , guessed value γ , and a vector $\mathbf{x} \in [0, 1]^n$.

Output: A vector $\mathbf{v} \in [0, 1]^n$.

- 1: For each $e \in E_S$, let $\boldsymbol{\theta}(e) = \text{ESTIMATE}_{\epsilon, \epsilon/n, \delta/n}(f_{R(\mathbf{x})}(e))$.
- 2: Find a vector $\mathbf{v} \in [0, 1]^n$ supported on E_S that minimizes $W(\mathbf{v})$ subject to

$$\mathbf{v} \cdot \boldsymbol{\theta} \geq (1 - \epsilon) \kappa' \gamma - \frac{\epsilon d}{\kappa'},$$

by the linear programming.

- 3: **return** \mathbf{v} .
-

Definition 3.1. *We say that γ is a good guess when the following fact holds:*

$$\mathbf{E} [f_{R(\mathbf{x})}(O_S)] \geq \gamma \geq (1 - \epsilon) \mathbf{E} [f_{R(\mathbf{x})}(O_S)] - \frac{\epsilon d}{\kappa'}.$$

Note that $\mathbf{E} [f_{R(\mathbf{x})}(O_S)] \leq \frac{1}{\kappa'} \mathbf{E} [f_{\emptyset}(O_S)] = \frac{1}{\kappa'} f(O_S) \leq \frac{1}{\kappa'^2} \sum_{e \in O_S} f(e) \leq \frac{1}{\kappa'^2} nd$, where the 2nd inequality is from Remark 2.1. So there must exist good guesses by enumerating all the values in $V_{\epsilon, n}/\kappa'^2 \triangleq \{v/\kappa'^2 | v \in V_{\epsilon, n}\}$. Then we conclude that the output vector \mathbf{v} of Algorithm 1 satisfies the conditions.

Lemma 3.3. *Suppose that γ is a good guess. Then, Algorithm 1 returns a vector $\mathbf{v} \in [0, 1]^n$ supported on E_S such that the following results hold with probability at least $1 - \delta$:*

1. $\sum_{e \in E_S} \mathbf{v}(e) \mathbf{E} [f_{R(\mathbf{x})}(e)] \geq (1 - \epsilon)^3 \kappa' \mathbf{E} [f_{R(\mathbf{x})}(O_S)] - \frac{3\epsilon d}{\kappa'}$,
2. $W(\mathbf{v}) \leq w(O_S)$.

Moreover, the oracle complexity of Algorithm 1 is $O(n^2 \log(n/\delta)/\epsilon^2)$.

3.2 Continuous Greedy Algorithm

The main algorithm in the continuous setting will be given in this section. Our expectation is that the outcome solution \mathbf{x} satisfies that $F(\mathbf{x}) \geq (1 - 1/e^{\kappa'^3})f(O)$ and $W(\mathbf{x}) \leq w(O)$. In previous sections, we assume that there are m elements in O_B . It should be noted that the definition of O_B depends on the choice of v , so does m . The following is a brief description on the idea of Algorithm 2.

In the outer loop we have the time variable $t \in [0, 1]$ increased from 0 to 1 at a constant rate of ϵ . Consequently, there are $\frac{1}{\epsilon}$ external iterations in total. In the inner loop of Algorithm 2, the vector $\mathbf{y}_i^{t+\epsilon} \in [0, 1]^n$, $\forall i \in [m]$ could be described as the direction selecting the i -th biggest marginal gain. Similarly, the vector $\mathbf{z}^{t+\epsilon}$ supported on E_S is generally to select those elements with enough marginal gains but small budgets. For every $t \in \{0, \epsilon, \dots, 1\}$, we update the current solution as

$$\mathbf{x}^{t+\epsilon} = \sum_{i \in [m]} \mathbf{y}_i^{t+\epsilon} + \mathbf{z}^{t+\epsilon}.$$

Algorithm 2. $\text{GCG}_{\epsilon, \delta}(f, w, E_B, E_S, m, \{\gamma_i^t\}, \{\gamma_S^t\})$

Input: A monotone κ' -submodular function $f : 2^E \rightarrow \mathbb{R}_+$, a weight function $w : E \rightarrow [0, 1]$, $\epsilon, \delta \in (0, 1)$, $E_B, E_S \subseteq E$, an integer $m \in \mathbb{N}$, guessed values $\{\gamma_i^t\}_{i \in [m], t \in \{0, \epsilon, \dots, 1 - \epsilon\}}$ and $\{\gamma_S^t\}_{t \in \{0, \epsilon, \dots, 1 - \epsilon\}}$.

Output: A vector $\mathbf{x}^1 \in [0, 1]^n$.

- 1: $\mathbf{y}_i^t \leftarrow \mathbf{0} \in [0, 1]^n$ for $i \in [m]$ and $\mathbf{z}^t \leftarrow \mathbf{0} \in [0, 1]^n$;
- 2: **for** ($t \leftarrow 0$; $t \leq 1 - \epsilon$; $t \leftarrow t + \epsilon$) **do**
- 3: **for** ($i \leftarrow 1$; $i \leq m$; $i \leftarrow i + 1$) **do**
- 4: $\theta_i^t(e) \leftarrow \text{ESTIMATE}_{\epsilon, \epsilon/m, \epsilon\delta/(2nm)} \left(E \left[f_{R(x_{i-1}^t)}(e) \right] \right)$ for each $e \in E$;
- 5: Let $e_i^t = \arg \min \{w(e) | \theta_i^t(e) \geq (1 - \epsilon)\gamma_i^t - \epsilon d / (m\kappa')\}$;
- 6: $\mathbf{y}_i^{t+\epsilon} \leftarrow \mathbf{y}_i^t + \epsilon \mathbf{1}_{e_i^t}$; $\mathbf{x}_i^t \leftarrow \sum_{j \leq i} \mathbf{y}_j^{t+\epsilon} + \sum_{j > i} \mathbf{y}_j^t + \mathbf{z}^t$;
- 7: **end for**
- 8: $\mathbf{v}^t \leftarrow \text{SMALL ELEMENTS}_{\epsilon, \epsilon\delta/2}(f, w, E_S, \gamma_S^t, \mathbf{x}_m^t)$;
- 9: $\mathbf{z}^{t+\epsilon} \leftarrow \mathbf{z}^t + \epsilon \mathbf{v}^t$, $\mathbf{x}^{t+\epsilon} \leftarrow \sum_{i \in [m]} \mathbf{y}_i^{t+\epsilon} + \mathbf{z}^{t+\epsilon}$;
- 10: **end for**
- 11: **return** \mathbf{x}^1 .

Similar to the treatment of small elements, we expect that the probability of the entire estimation step being satisfactory at the same time is not less than $1 - \delta/2$, whose sufficient condition is each satisfying estimation has probability $1 - \epsilon\delta/(2nm)$. Let us start with estimation $\mathbf{E} \left[f_{R(x_{i-1}^t)}(e) \right]$ by $\theta_i^t(e)$.

Lemma 3.4. *With probability at least $1 - \epsilon\delta/(2nm)$, we have*

$$(1 - \epsilon)\mathbf{E} \left[f_{R(x_{i-1}^t)}(e) \right] - \frac{\epsilon d}{m\kappa'} \leq \theta_i^t(e) \leq (1 + \epsilon)\mathbf{E} \left[f_{R(x_{i-1}^t)}(e) \right] + \frac{\epsilon d}{m\kappa'},$$

for each $t \in \{0, \epsilon, \dots, 1 - \epsilon\}$, $i \in [m]$, and $e \in E$.

We formalize the concept that $\{\gamma_i^t\}$ are sufficiently accurate.

Definition 3.2. For $t \in \{0, \epsilon, \dots, 1 - \epsilon\}$ and $i \in [m]$, we say that γ_i^t is a good guess if

$$\mathbf{E} \left[f_{R(x_{i-1}^t)}(o_i) \right] \geq \gamma_i^t \geq (1 - \epsilon) \mathbf{E} \left[f_{R(x_{i-1}^t)}(o_i) \right] - \frac{\epsilon d}{m},$$

holds.

Since $\mathbf{E} \left[f_{R(x_{i-1}^t)}(o_i) \right] \leq \frac{1}{\kappa'} \mathbf{E} [f_\emptyset(o_i)] = \frac{1}{\kappa'} f(o_i) = \frac{1}{\kappa'} f(o_i) \leq \frac{1}{\kappa'} d$, we can find good guesses by trying all the values in the set $V_{\epsilon, m} / (m\kappa') := \{v / (m\kappa') \mid v \in V_{\epsilon, m}\}$.

Lemma 3.5. Suppose that the consequence of Lemma 3.4 holds and that $\{\gamma_i^t\}$ are good guesses. Then, for every $t \in \{0, \epsilon, \dots, 1 - \epsilon\}$ and $i \in [m]$, we have the following results:

1. $\mathbf{E} \left[f_{R(x_{i-1}^t)}(e_i^t) \right] \geq (1 - \epsilon)^3 \mathbf{E} \left[f_{R(x_{i-1}^t)}(o_i) \right] - \frac{3\epsilon d}{m\kappa'}$,
2. $w(e_i^t) \leq w(o_i)$.

Lemma 3.6. Let $f : 2^E \rightarrow \mathbb{R}_+$ be a monotone and κ' -submodular function, given a positive real ϵ and vectors $\mathbf{x}, \mathbf{y} \in [0, 1]^n$ such that $\mathbf{x} + \epsilon \mathbf{y} \in [0, 1]^n$. We show that

$$F(\mathbf{x} + \epsilon \mathbf{y}) - F(\mathbf{x}) \geq \epsilon \kappa' \sum_{e \in E} \mathbf{y}(e) \mathbf{E} \left[f_{R(\mathbf{x} + \epsilon \mathbf{y})}(e) \right].$$

The theorem below explains the approximation ratio of the fractional solution.

Theorem 3.1. Suppose that $\{\gamma_i^t\}$ and $\{\gamma_S^t\}$ are all good guesses. Then, Algorithm 2 returns a vector \mathbf{x}^1 satisfying

1. $F(\mathbf{x}^1) \geq (1 - 1/e^{\kappa'^3} - O(\epsilon))f(O) - 6\epsilon d / \kappa'^4$,
2. $W(\mathbf{x}^1) \leq w(O)$,

with probability at least $1 - \delta$. The oracle complexity is $O\left(\frac{nm^2}{\epsilon^3} \log \frac{nm}{\epsilon\delta} + \frac{n^2}{\epsilon^3} \log \frac{n}{\epsilon\delta}\right)$.

4 Rounding by the CR Scheme

In this section, we will show the CR scheme for rounding fractional solutions. Consider a knapsack polytope $P_{\mathcal{F}} = \text{conv} \{\mathbf{1}_F : \sum_{e \in F} w(e) \leq 1, F \subseteq E\}$. For any vector $\mathbf{x} = (x_1, \dots, x_n) \in P_{\mathcal{F}}$, we recall that $R(\mathbf{x})$ is a random subset such that each element $j \in E$ is independently included with probability x_j . Thus, $\Pr[R(\mathbf{x}) = R] = \prod_{j \in R} x_j \prod_{j \notin R} (1 - x_j)$. The random set $R(\mathbf{x})$ does not always satisfy the knapsack constraint. In order to obtain a feasible set $I \subseteq R(\mathbf{x})$, we use the CR scheme to remove some elements from $R(\mathbf{x})$. As an actual fact, there are two random events that occur in sequence. Initially we pick a random set $R(\mathbf{x})$ from E , and next pick a random set I from the random set $R(\mathbf{x})$.

Definition 4.1 (a (b, c) -balanced CR scheme, [4]). A (b, c) -balanced CR scheme for $P_{\mathcal{F}}$ is a scheme such that for any $\mathbf{x} \in P_{\mathcal{F}}$, the scheme selects a feasible subset $I \subseteq R(\mathbf{bx})$ with the following property: $\Pr[i \in I | i \in R(\mathbf{bx})] \geq c$ for every element i . The scheme is said to be monotone if $\Pr[i \in I | R(\mathbf{bx}) = R_1] \geq \Pr[i \in I | R(\mathbf{bx}) = R_2]$ whenever $i \in R_1 \subseteq R_2$. A scheme is said to be strict if $\Pr[i \in I | i \in R(\mathbf{bx})] = c$ for every i .

The following lemma shows that a $(1 - \varepsilon, 1 - e^{-\Omega(\varepsilon^2/\lambda)})$ -balanced CR scheme exists for a knapsack constraint.

Lemma 4.1 (Lemma 4.14, [4]). For any $\lambda, \varepsilon > 0$ and a knapsack constraint $\mathcal{F} = \{S : \sum_{i \in S} w_i \leq 1\}$ such that $w_i \leq \lambda$ for all i , there is a monotone $(1 - \varepsilon, 1 - e^{-\Omega(\varepsilon^2/\lambda)})$ -balanced contention resolution scheme.

The main problem then becomes the relationship between the feasible set I from CR scheme and our continuous solution \mathbf{x} . The fractional solution \mathbf{x} together with the (b, c) -balanced CR scheme imply a random set $R(\mathbf{bx})$. Then a feasible set I can be obtained by CR scheme with high probability. What follows is an exploration between the two terms $F(\mathbf{bx})$ and $\mathbf{E}[f(I)]$. In the proof of the main rounding theorem, we will use the variant of an important inequality.

Lemma 4.2 (Application of FKG inequality). Let $X = 2^E$ and $i \in E$. Let π be a (b, c) -balanced CR scheme for $P_{\mathcal{F}}$ and $\mathbf{x} \in P_{\mathcal{F}}$. For any subset $R \subseteq E$ and any fixed element $i \in E$, define $g(R) = \mathbf{E}_{I \leftarrow \pi_{\mathbf{bx}}(R)}[\mathbf{1}_{i \in I} | R]$, $h(R) = L(R, i)$, $\mu(R) = \Pr[R | i \in R]$, where $\mathbf{1}_{i \in I}$ equals to 1 if $i \in I$ and 0 otherwise. Then

$$\sum_{R \in X} g(R)h(R)\mu(R) \geq \left(\sum_{R \in X} g(R)\mu(R) \right) \left(\sum_{R \in X} h(R)\mu(R) \right).$$

The following lemma is a necessary conclusion in the rounding routine.

Lemma 4.3. Suppose $b \in (0, 1]$ and $\mathbf{x} \in [0, 1]^n$. Assume F is the multilinear extension of a set function f with continuous generic submodularity ratio κ . It follows that

$$F(\mathbf{bx}) \geq \frac{1}{1 + \frac{1-b}{\kappa^2 b}} F(\mathbf{x}).$$

When some definitions and lemmas are ready, we now present the following theorem.

Theorem 4.1. Suppose that $P_{\mathcal{F}}$ is a knapsack constraint polytope, and π is a (b, c) -balanced CR scheme over $\mathbf{x} \in P_{\mathcal{F}}$, then we have

$$\mathbf{E}_{\substack{R \leftarrow R(\mathbf{bx}) \\ I \leftarrow \pi_{\mathbf{bx}}(R)}} [f(I)] \geq c\kappa F(\mathbf{bx}).$$

Recall that Lemma 4.1 provides us a monotone $(1 - \varepsilon, 1 - e^{-\Omega(\varepsilon^2/\lambda)})$ -balanced CR scheme, therefore we immediately get the following corollary.

Corollary 4.1. *Suppose that $P_{\mathcal{F}}$ is a knapsack constraint polytope, and π is a (b, c) -balanced CR scheme over $\mathbf{x} \in P_{\mathcal{F}}$, then we get*

$$\mathbf{E}_{\substack{R \leftarrow R(b\mathbf{x}) \\ I \leftarrow \pi_{b\mathbf{x}}(R)}} [f(I)] \geq \kappa \left(1 - e^{-\Omega(\varepsilon^2/\lambda)}\right) F((1 - \varepsilon) \mathbf{x}).$$

Together with Lemma 4.3, we can obtain the following result.

Theorem 4.2. *Suppose that $P_{\mathcal{F}}$ is a knapsack constraint polytope, and π is a (b, c) -balanced CR scheme over $\mathbf{x} \in P_{\mathcal{F}}$, then it holds*

$$\mathbf{E}_{\substack{R \leftarrow R(b\mathbf{x}) \\ I \leftarrow \pi_{b\mathbf{x}}(R)}} [f(I)] \geq \kappa \left(\frac{1}{1 + \frac{\varepsilon}{\kappa^2(1-\varepsilon)}} \right) \left(1 - e^{-\Omega(\varepsilon^2/\lambda)}\right) F(\mathbf{x}).$$

5 The Whole Procedure

The following is a brief description of the entire algorithm. The idea is to simply guess $v, m, \{\gamma_i^t\}, \{\gamma_S^t\}$, run continuous greedy algorithm with the guessed values from enumeration, and then round the obtained vectors \mathbf{x} using the CR scheme. A detailed form of our procedure is given in Algorithm 3.

Algorithm 3. KNAPSACK

Input: A monotone function $f : 2^E \rightarrow \mathbb{R}_+$ with continuous generic submodularity ratio κ and generalized curvature τ , a weight function $w : E \rightarrow [0, 1]$, and $\varepsilon \in (0, 1)$.

Output: A set $S \subseteq E$ satisfying $w(S) \leq 1$ with high probability.

```

1: for each choice of  $v \in V_{\varepsilon, n/\kappa}$  do
2:    $E_B \leftarrow$  the set of big elements with respect to  $v$ ;
3:    $E_S \leftarrow$  the set of small elements with respect to  $v$ ;
4:    $\mathcal{S} \leftarrow \emptyset$ ;
5:    $M := \lfloor \frac{1}{(1-\tau)\varepsilon^6} \rfloor$ ;
6:   for each choice of  $m$  from  $\{0, 1, \dots, M\}$  do
7:     for each choice of  $\{\gamma_i^t\}$  from  $V_{\varepsilon, m/m\kappa}$  do
8:       for each choice of  $\{\gamma_S^0, \gamma_S^\varepsilon, \dots, \gamma_S^{1-\varepsilon}\}$  from  $V_{\varepsilon, n/\kappa^2}$  do
9:          $\text{GCG}_{\varepsilon, \varepsilon}(f, w, E_B, E_S, m, \{\gamma_i^t\}, \{\gamma_S^t\})$ ;
10:         $S \leftarrow$  the CR scheme $_{\varepsilon}(E_B, E_S; \mathbf{x})$ ;
11:         $\mathcal{S} \leftarrow \mathcal{S} \cup \{S\}$ ;
12:       end for
13:     end for
14:   end for
15: end for
16: return  $\arg \max_{S \in \mathcal{S}} f(S)$ .
```

Theorem 5.1. *Given a monotone function $f : 2^E \rightarrow \mathbb{R}_+$ with continuous generic submodularity ratio κ , a weight function $w : E \rightarrow [0, 1]$, and $\epsilon \in (0, 1)$, Algorithm 3 outputs a (random) set $I \subseteq E$ and $w(I) \leq 1$ with probability $(1 - e^{-\Omega(\epsilon^2/\lambda)})$ satisfying*

$$\mathbf{E}[f(I)] \geq \kappa \left(\frac{1}{1 + \frac{\epsilon}{\kappa^2(1-\epsilon)}} \right) \left(1 - e^{-\Omega(\epsilon^2/\lambda)} \right) \left(1 - e^{-\kappa^3} - O(\epsilon) \right) f(O),$$

where λ denotes the largest weight. The oracle complexity is

$$O \left(\left(\frac{n}{(1-\tau)^3} \frac{1}{\epsilon^{21}} \log \frac{n}{\epsilon} + \frac{1}{1-\tau} \frac{n^2}{\epsilon^9} \log \frac{n}{\epsilon} \right) \cdot \left(\frac{\log(n/\epsilon)}{\epsilon} \right)^{\left(\frac{1}{\epsilon}+1\right)} \cdot \left(\frac{\log(\frac{1}{\epsilon})}{\epsilon} \right)^{\frac{1}{1-\tau} \frac{1}{\epsilon^7}} \right).$$

When the function f is submodular, we have $\kappa = 1$. Then we obtain the following corollary.

Corollary 5.1. *Given a monotone submodular function $f : 2^E \rightarrow \mathbb{R}_+$, a weight function $w : E \rightarrow [0, 1]$, and $\epsilon \in (0, 1)$, Algorithm 3 outputs a (random) set $I \subseteq E$ and $w(I) \leq 1$ with probability $(1 - e^{-\Omega(\epsilon^2/\lambda)})$ satisfying*

$$\mathbf{E}[f(I)] \geq (1 - \epsilon) \left(1 - e^{-\Omega(\epsilon^2/\lambda)} \right) (1 - e^{-1} - O(\epsilon)) f(O),$$

where λ denotes the largest weight.

References

1. Badanidiyuru, A., Vondrák, J.: Fast algorithms for maximizing submodular functions. In: Proceedings of SODA, pp. 1497–1514 (2013). <https://doi.org/10.1137/1.9781611973402.110>
2. Bian, A.A., Buhmann, J.M., Krause, A., Tschischek, S.: Guarantees for greedy maximization of non-submodular functions with applications. In: Proceedings of ICML, pp. 498–507 (2017). <https://doi.org/10.5555/3305381.3305433>
3. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a submodular set function subject to a matroid constraint (extended abstract). In: Fischetti, M., Williamson, D.P. (eds.) IPCO 2007. LNCS, vol. 4513, pp. 182–196. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72792-7_15
4. Chekuri, C., Vondrák, J., Zenkluse, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM J. Comput.* **43**(6), 1831–1879 (2014)
5. Ene, A., Nguyễn, H.L.: A nearly-linear time algorithm for submodular maximization with a knapsack constraint. In: Proceedings of ICALP, pp. 53:1–53:12 (2019). <https://doi.org/10.4230/LIPICs.ICALP.2019.53>
6. Feldman, M.: Maximization problems with submodular objective functions. Ph.D. thesis, Computer Science Department, Technion (2013)
7. Fortuin, C.M., Kasteleyn, P.W.: Correlation inequalities on some partially ordered sets. *Commun. Math. Phys.* **22**(2), 89–103 (1971)

8. Gong, S., Nong, Q., Liu, W., Fang, Q.: Parametric monotone function maximization with matroid constraints. *J. Global Optim.* **75**(3), 833–849 (2019)
9. Kleywegt, A.J., Papastavrou, J.D.: The dynamic and stochastic knapsack problem. *Oper. Res.* **46**(1), 17–35 (1998)
10. Mansini, R., Speranza, M.G.: A multidimensional knapsack model for asset-backed securitization. *J. Oper. Res. Soc.* **53**(8), 822–832 (2002)
11. Shamir, A.: A polynomial time algorithm for breaking the basic Merkle-Hellman cryptosystem. In: *Proceedings of SFCS*, pp. 145–152 (1982). <https://doi.org/10.1109/SFCS.1982.5>
12. Sviridenko, M.: A note on maximizing a submodular set function subject to a knapsack constraint. *Oper. Res. Lett.* **32**(1), 41–43 (2004)
13. Sviridenko, M., Vondrák, J., Ward, J.: Optimal approximation for submodular and supermodular optimization with bounded curvature. *Math. Oper. Res.* **42**(4), 1197–1218 (2017)
14. Yoshida, Y.: Maximizing a monotone submodular function with a bounded curvature under a knapsack constraint. *SIAM Discrete Math.* **33**(3), 1452–1471 (2019)
15. Zhang, Z., Liu, B., Wang, Y., Xu, D., Zhang, D.: Greedy algorithm for maximization of non-submodular functions subject to knapsack constraint. In: Du, D.-Z., Duan, Z., Tian, C. (eds.) *COCOON 2019. LNCS*, vol. 11653, pp. 651–662. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26176-4_54
16. Zhang, Z., Liu, B., Wang, Y., Xu, D., Zhang, D.: Greedy algorithm for maximization of non-submodular functions subject to knapsack constraint. In: Du, D.-Z., Duan, Z., Tian, C. (eds.) *COCOON 2019. LNCS*, vol. 11653, pp. 651–662. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26176-4_54



FEENET: A Real-Time Semantic Segmentation via Feature Extraction and Enhancement

Sixiang Tan^{1,2}, Wenzhong Yang^{1,2}(✉), JianZhuang Lin^{1,2}, and Weijie Yu^{1,2}

¹ College of Information Science and Engineering, Xinjiang University,
Urumqi, Xinjiang, China
ywz_xy@163.com

² Key Laboratory of Multilingual Information Technology in Xinjiang Uygur
Autonomous Region, Xinjiang University, Urumqi, Xinjiang, China

Abstract. As a pixel-level prediction task, semantic segmentation need rich spatial information. However, most popular real-time network architectures tend to compromise spatial resolution to increase speed, but the accuracy is greatly reduced as a result. Therefore, we propose a novel Feature Enhancement Module (FEM) to extracted and enhanced the future map before the image down-sample on the backbone. Meanwhile, since the low-layer have rich detail information and high-level contain more semantic information, we propose a Feature Extraction and Fusion Module (FEFM) to fuse low-level and high-level feature representation. Based on the FEM and FEFM, we introduce a real-time semantic segmentation network FEENET. Experiments on Cityscapes and CamVid datasets demonstrate that the proposed FEENET achieves a balance between speed computation and accuracy. Without additional processing and pre-training, it achieves 75.47% Mean IoU on the Cityscapes test dataset with only 4.35G Flops and a speed of 94 FPS on a single RTX 2080Ti card. Code is available at <https://github.com/favoMJ/FEENet>.

Keywords: Semantic segmentation · Feature enhancement · Real-time network · Feature extraction

1 Introduction

Image classification, target detection, and semantic segmentation are the three basic tasks of computer vision. Among them, image classification is to determine the classification of the content in the image. Target detection is to identify the content of the image, but also to give the location information of the target. Semantic segmentation aims to assign dense labels for all pixels. Although many previous works have made great progress in accuracy [1–3], the time used, the amount of calculation, and the memory consumed are all subject to limitations in practical applications. Therefore, a light-weight, effective and high accuracy model is very necessary. Most of the previous excellent work [4–6] has already

proved the availability of down-sample operations, but this will cause the loss of semantic information. Some methods [7, 8] is to obtain both semantic information and spatial information through dual branches. Some works [9, 10] is to reduce the information loss during down-sampling or seek better feature fusion low-layer details feature and high-layer representation feature. Therefore, we hold the opinions that a combination of well-designed semantic information extraction and information fusion technology is especially suitable for real-time semantic segmentation.

Based on the above observation, we propose a novel network architecture specially designated for real-time semantic segmentation, which is presented in Fig. 1. We develop a light-weighted Feature Enhancement Module (FEM) to extract and enhance semantic information, and we proposed a powerful Feature Extraction and Fusion Module (FEFM) to fusion shallow feature image information with deep features.

Our main contributions are summarized as follows:

- We design a light-weight and powerful backbone with feature extraction function to enhance the entire network.
- We elaborately design two specific modules, Feature Enhancement Module (FEM) and Feature Extraction and Fusion Module (FEFM) to provide comparable accuracy at an acceptable cost.
- We achieve impressive results on the benchmarks of Cityscapes [28] and CamVid [29] benchmarks without any pretrained model. It can run on high-resolution images (512 1024) at 94 FPS on a single RTX 2080Ti card and obtain the results of 75.47% mean Intersection over Union (mIoU%) on the Cityscapes test dataset.

2 Relate Work

Recently, many new methods have been used in semantic segmentation. In this section, we introduce 4 groups of methods related to our work. i.e., real-time semantic segmentation methods, feature fusion, feature enhancement, and light-weight architectures.

2.1 Real-Time Semantic Segmentation

The main purpose of real-time segmentation is to obtain high-quality predictions while obtaining faster inference speed. ESPNetv2 [11] uses group point-wise and depth-wise dilated separable convolutions to learn representations from a large effective receptive field with fewer FLOPs and parameters. ICNet [12] incorporates multi-resolution branches under proper label guidance. SegNet [4] present a deep fully convolutional neural network architecture for semantic pixel-wise segmentation. ENet [13] trims a great number of convolution filters to reduce computation. Different from the models mentioned above, our proposed method Feature Enhancement Module employs a low-pass filtering to save more information at a faster inference speed.

2.2 Feature Fusion

Traditional information collection methods are usually just through sum up features. RefineNet [14] introduces a complicated refine module in each upsampling stage between the encoder and decoder to extract multi-scale features. BiSenet [7] propose a feature fusion module to fuse these features. DFANet [9] aggregation strategy is composed of subnetwork aggregate and substage aggregate methods aims to exploit features combined from both network-level and stage-level. MSFNet [15] uses feature fusion extensively to improve the interaction between different layers in terms of spatial information and semantic information. Our model uses feature fusion extensively to improve the interaction between different layers in terms of spatial information and semantic information, which improves the spatial sensitivity of the network obviously aph headings as needed.

2.3 Feature Enhancement

The accuracy of semantic segmentation is inseparable from rich feature information. CCNet [16] propose a novel criss-cross attention module in this work, which can be leveraged to capture contextual information from long-range dependencies in a more efficient and effective way. AaA [17] propose an adaptive content-aware low-pass filtering layer. SIA [18] integrate classic anti-aliasing filtering to improve shift-equivariance/invariance of deep networks. Our mode uses anti-aliasing filtering to extract important part of the feature.

2.4 Light-Weight Architectures

Light-weight models save storage space and memory, also reduces the amount of calculation. CGNet [19] propose a CG block to effectively and efficiently capture contextual information in all stages. ERFNet [20] uses residual connections and factorized convolutions in order to remain efficient while retaining accuracy. LEDNet [21] adopting an asymmetric encoder-decoder architecture for real-time semantic segmentation. Our model follows the light-weight style to achieve real-time segmentation.

3 The Proposed Method

In this section, we will elaborate on our proposed network. We first introduce our Feature-extraction model in detail. The whole architecture of Feature Extraction and Enhancement Network (FEENet) is shown in Fig. 1.

3.1 Feature Enhancement Module

Semantic segmentation requires needs rich detail information and semantic information, where high-level features are rich in lack semantic information but lack in spatial details and vice versa. Unlike MFENet [34] combine the Laplace operator and the convolution kernels to extract spatial information from low-level

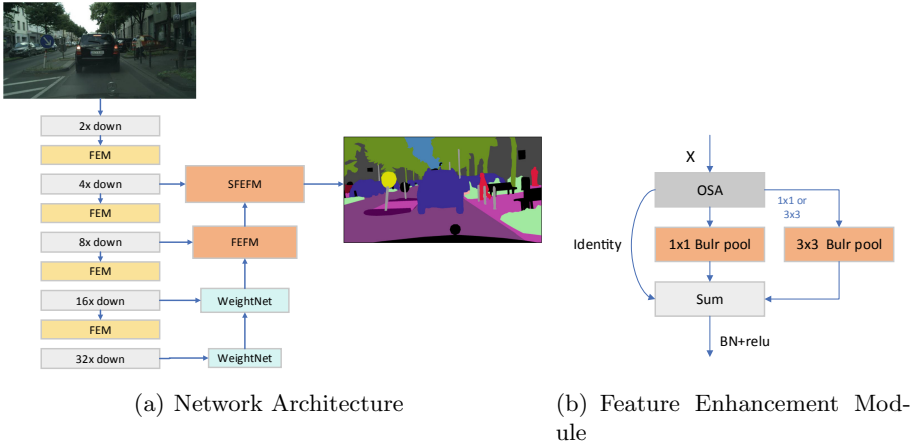


Fig. 1. An overview of the feature extraction and enhancement network. (a) Network Architecture. (b) Feature Enhancement Module (FEM), Bulr pool represents low-pass filtering, BN denotes Batch Normalization [25].

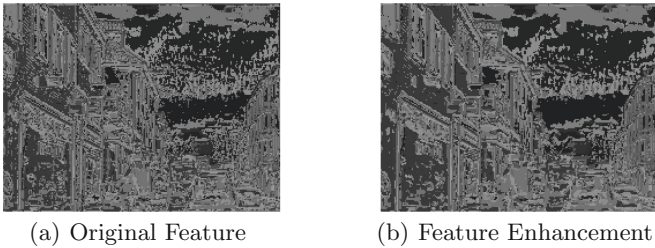


Fig. 2. An Comparison between of the original Feature and Feature Enhancement. (a) Original Feature. (b) Feature Enhancement, have stronger information expression ability.

features, we try to use low-pass filtering to enhance semantic information and suppress noise at the low-level features. We think that the rich semantic information is helpful to the boundary feature extraction to a certain extent, so we proposed Feature Enhancement Module (FEM) to bridge the gap between low-level and high-level features, as illustrated in Fig. 1(b). Different from the usual segmented network use the backbone like ResNet [22] or Xception [23]. We combine OSA [24] with FEM as our backbone for further boosting the performance of EFENet. The enhanced contrast is shown in Fig. 2.

To exploit the feature extraction map $F_e(X_i) \in R^{c \times w \times h}$ as a feature descriptor map $X_i \in R^{c \times w \times h}$, the FEM generates a low-pass filtering to reduce the noise. K_{lf} The computation process is summarized as follow:

$$F_e(X_i) = K_{lf}(X_i) \tag{1}$$

Finally, the output feature $X_{output} \in R^{c \times w \times h}$ is computed as:

$$X_{output} = X + F_e(X_i) \quad (2)$$

3.2 Spatial Path and Context Path

Semantic segmentation requires enough semantic information and rich detail information. If properly trained, deep CNN can capture rich scene information through multilayer convolution operations and nonlinear pooling/activation functions, however, approaches like atrous spatial pyramid pooling have large amount of calculation and memory usage, resulting in low speed.

We use the two branch architecture like BiSeNet the difference is that we not use three layers to obtain spatial information. Instead, after the feature is enhanced, the spatial feature is obtained through information filtering. In last two stage, we deploy WeightNet [27] to refine the features and U-shape structure [4] to fuse the features.

3.3 Feature Extraction and Fusion Module

In many works of deep learning, fusing features of different scales is an important means to improve performance. The low-level features have a higher resolution and contain more position and detail information, but due to fewer convolutions, they have lower semantics and more noise. High-level features have stronger semantic information, but the detail is lost.

We propose the Feature Extraction and Fusion module, as illustrated in Fig. 3(a). We reduce the cost through Conv1 and Conv2 to reduced channels for spatial aware feature $F_s \in R^{c \times w \times h}$ and content path output feature $F_c \in R^{c \times w \times h}$. To extract the feature of the $F_e(X_i)$, first normalize the F_c by sigmoid then multiply with F_s . The process is defined as:

$$F_{e1}(X_i) = \sigma(F_c) \otimes F_s \quad (3)$$



(a) Feature Extraction and Fusion Module (FEFM), (b) Small Feature Extraction and Fusion Module (SFEFM)

Fig. 3. FFEM and SFEM. (a) Feature Extraction and Fusion Module (FEFM), (b) Small Feature Extraction and Fusion Module (SFEFM), \otimes represents element-wise product.

$$F_{e2}(X_i) = F_c \otimes F_s \quad (4)$$

Where σ denotes the sigmoid function and \otimes represents element-wise multiplication. Finally, the enhance feature F_{output} is computed as:

$$F_{output} = ConvBN_{3 \times 3}(F_{e1}(X_i) + F_{e2}X_i) \quad (5)$$

$ConvBN_{3 \times 3}$ denotes 3×3 conv layer.

Moreover, In order to obtain more detail information without increasing the cost, we propose Small Feature Extraction and Fusion Module(SFEM). The process is defined as:

$$F_{output} = up(F_{input}) + \sigma(F_{detail}) * (F_{detail}) \quad (6)$$

where σ denotes the sigmoid function.

4 Experiments

We evaluate our proposed network on Cityscapes [28] and Camvid [29]. We first introduce the datasets and implementation protocol. Then, we perform a series of ablation experiments on Camvid validation dataset to prove the effectiveness of our network. Finally, we carry out comprehensive experiments on Cityscapes and CamVid benchmarks and compare with the state-of-the-art works, we use the standard mIoU metric to report segmentation accuracy.

4.1 Datasets and Settings

Cityscapes. Cityscapes has 5000 images of driving scenes in an urban environment (2975 train, 500 val, 1525 test). It has 19 categories of dense pixel annotation (97% coverage), of which 8 have instance-level segmentation. The Cityscapes data set, the urban landscape data set, is a large-scale data set that contains a set of different stereo video sequences recorded in 50 different city street scenes.

Camvid. CamVid is the first video collection with semantic tags for the target category. It includes a total of 701 images, 367 for training, 101 for validation and 233 for testing. The image has a resolution of 360480 and 11 semantic categories.

Implementation Protocol. All the experiments are performed with one 2080Ti GPU, CUDA 10.1 on the Pytorch platform. Runtime evaluation is performed on a single 2080Ti card, we report an average of 100 frames for the frames per second (FPS) measurement.

The Adam optimizer [11] is adopted to train our model. Specifically, the batch size is set to 6. We use cosine attenuation [15] with initial learning rate to $1e^{-4}$ and a minimum learning rate to $1e^{-6}$. We train the model for 600 epochs on Cityscapes dataset, and Camvid set 1000 epochs on Camvid. For data augmentation, we employ random horizontal flip and mean subtraction. We randomly use the parameters between [0.5, 2] to transform the image to different scales, and then we randomly crop the resolution to 5121024 on Cityscapes for training while the cropping resolution is 360480 on Camvid.

4.2 Ablation Studies

Table 1. Ablation study results on cityscapes val set. FPS are estimated for an input of 5121024 on a single RTX 2080Ti

Network	mIoU (%)	BaseModel	FPS
FEENet	72.1	Resnet18	110
FEENet	74.5	Vov19-Slim + FFM	101
FEENet	75.1	Vov19-Slim + FEFM	103
FEENet	76.8	Vov19-Slim + FEFM + FEM	96
FEENet	78.2	Vov19-Slim + FEFM + SFEFM + FEM	94

To further prove the effectiveness of the FEENET, we conduct extensive ablation experiments on the validation set of Cityscapes with different settings for FEENET.

Feature Enhancement Module. For striking a balance between the speed and prediction performance, we have added the FFM module to Vovnet as our backbone. As can be seen in Table 1, the Feature Enhancement Module(FEM) can bring in over 1.7% improvement to the cityscapes val dataset.

Feature Extraction and Fusion Module. For further improving the performance, we specially design a Feature Extraction and Fusion Module(FEFM). As can be seen from Table 1, adding the FEFM can bring 0.6% slightly better accuracy than FFM [7] while resulting in increased runtime with only 101 FPS, adding the SFEFM can bring 1.7% better accuracy.

Table 2. Evaluation results on the Cityscapes test set

Model	Input size	FLOPs	FPS	GPU	mIoU (%)
SegNet [4]	640360	286G	16.7	TitanX M	57.0
ENet [13]	640360	3.8G	135.4	TitanX M	57.0
SQ [5]	10242048	270G	16.7	TitanX	59.8
CGNET [19]	10242048	14.8	50.0	Tesla K80	64.8
ERFNet [20]	5121024	21.0G	41.7	TitanX M	68.0
ICNet [12]	10242048	28.3G	30.3	TitanX M	69.0
BiSeNetV2 [7]	768 × 1536	55.3G	45.7	TitanX	74.7
DABNet [30]	10242048	10.46G	58.04	1080Ti	70.1
LADNet [33]	5121024	11.5G	71	1080Ti	69.2
BiSeNet [7]	1024 × 2048	103.72G	47.20	2080Ti	74.7
ShelfNet [35]	1024 × 2048	93.69G	44.37	2080Ti	74.8
Ours	5121024	4.35G	94.0	2080Ti	75.4

4.3 Comparison with State-of-the-Art Works

In this subsection, we compare our algorithm with state-of-the-art models. Comparison of the accuracy and speed of our models is shown in Table 2. During evaluation, we do not adopt any testing tricks such as multi-crop and multi-scale testing. Our experimental environment is a single RTX 2080 Ti GPU. Some visual results of the proposed FEENET are shown in Fig. 4. We can achieve high performance of semantic segmentation on Cityscapes using our proposed Feature Extraction and Enhancement Network.

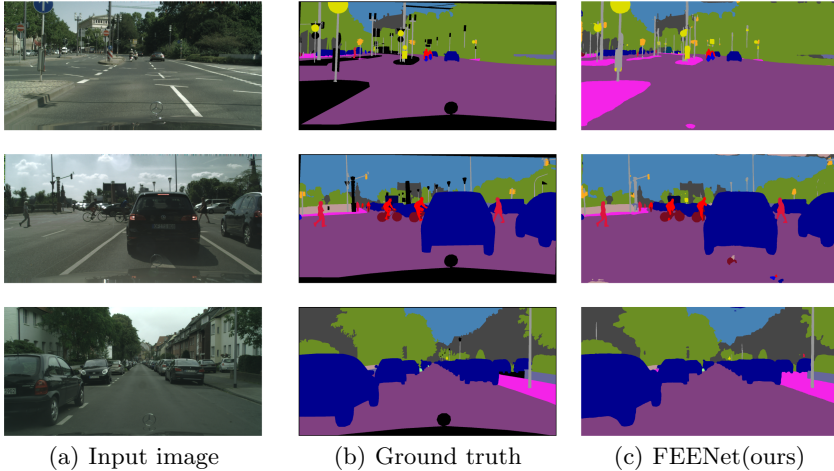


Fig. 4. Qualitative examples of the segmentation on Cityscapes validation set. From left to right: Input image, ground-truth, and prediction of FEENet.

As can be observed in Table 2, while the inference speed of the proposed method significantly outperforms state-of-the-art methods, the accuracy performance is kept comparable, attributing to the simple and efficient pipeline. The baseline of the proposed method achieves mIoU 75.4% on Cityscapes test set with 94 FPS inference speed.

4.4 Comparison on Other Datasets

We also evaluate our network on CamVid dataset. As shown in Table 3, our model achieves outstanding performance with small capacity, and it can process a 360480 CamVid image at the speed of 225 FPS.

Table 3. A Evaluation results on the CamVid test set

Model	FPS	mIoU (%)
DPN [31]	1.2	60.1
DeepLab [32]	4.9	61.6
ENet [13]	–	51.3
ICNet [12]	27.8	67.1
BiSeNet [7]	–	65.6
DFANet A [9]	120	64.7
DFANet B	160	59.3
Ours	225	68.1

5 Conclusion

In this paper, we have described a FEENet model to tackle real-time semantic segmentation. The backbone use a feature enhancement module, enhancing information communication in the manner of feature reuse. On the other hand, the decoder employs a feature extraction and fusion module, Make full use of low-level and high-level semantic information. The experimental results show our FEENet achieves best trade-off on CityScapes dataset in terms of segmentation accuracy and implementing efficiency. The future work includes improve the receptive field without adding additional calculations, resulting in further light-weight network while still remaining segmentation accuracy.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (No. U1603115), National key R&D plan project (2017YF C0820702-3) and National Engineering Laboratory for Public Safety Risk Perception and Control by Big Data (XJ201810101).

References

1. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
2. Fu, J., Liu, J., Tian, H., et al.: Dual attention network for scene segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3146–3154 (2019)
3. Huang, Z., Wang, X., Wei, Y., et al.: CCNet: criss-cross attention for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **PP**(99), 1 (2020)
4. Badrinarayanan, V., Kendall, A., Cipolla, R.: SegNet: a deep convolutional encoder-decoder architecture for image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**, 2481–2495 (2017)
5. Treml, M., et al.: Speeding up semantic segmentation for autonomous driving. In: MLITS, NIPS Workshop (2016)

6. Wu, Z., Shen, C., van den Hengel, A.: Real-time semantic image segmentation via spatial sparsity. arXiv preprint [arXiv:1712.00213](https://arxiv.org/abs/1712.00213) (2017)
7. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: BiSeNet: bilateral segmentation network for real-time semantic segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11217, pp. 334–349. Springer, Cham (2018a). https://doi.org/10.1007/978-3-030-01261-8_20
8. Wang, L.W., Siu, W.C., Liu, Z.S., et al.: Deep relighting networks for image light source manipulation. arXiv preprint [arXiv:2008.08298](https://arxiv.org/abs/2008.08298) (2020)
9. Li, H., Xiong, P., Fan, H., Sun, J.: DFANet: deep feature aggregation for real-time semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 4, p. 13 (2019b)
10. Wang, P., Chen, P., Yuan, Y., et al.: Understanding convolution for semantic segmentation. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1451–1460. IEEE (2018)
11. Mehta, S., Rastegari, M., Shapiro, L.G. Hajishirzi, H.: ESPNetv2: a light-weight, power efficient, and general purpose convolutional neural network. CoRR, abs/1811.11431 (2018)
12. Zhao, H., Qi, X., Shen, X., Shi, J., Jia, J.: ICNet for real-time semantic segmentation on high-resolution images. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11207, pp. 418–434. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01219-9_25
13. Paszke, A., Chaurasia, A., Kim, S., et al.: ENet: a deep neural network architecture for real-time semantic segmentation (2016)
14. Lin, G., Milan, A., Shen, C., et al.: RefineNet: multi-path refinement networks for high-resolution semantic segmentation (2016)
15. Si, H., Zhang, Z., Lv, F., et al.: Real-time semantic segmentation via multiply spatial fusion network. arXiv preprint [arXiv:1911.07217](https://arxiv.org/abs/1911.07217) (2019)
16. Huang, Z., Wang, X., Huang, L., et al.: CCNet: criss-cross attention for semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 603–612 (2019)
17. Zou, X., Xiao, F., Yu, Z., et al.: Delving deeper into anti-aliasing in ConvNets. arXiv preprint [arXiv:2008.09604](https://arxiv.org/abs/2008.09604) (2020)
18. Zhang, R.: Making convolutional networks shift-invariant again. arXiv preprint [arXiv:1904.11486](https://arxiv.org/abs/1904.11486) (2019)
19. Wu, T., Tang, S., Zhang, R., et al.: CGNet: a light-weight context guided network for semantic segmentation. arXiv preprint [arXiv:1811.08201](https://arxiv.org/abs/1811.08201) (2018)
20. Romera, E., Alvarez, J.M., Bergasa, L.M., et al.: ERFNet: efficient residual factorized ConvNet for real-time semantic segmentation. *IEEE Trans. Intell. Transp. Syst.* **19**(1), 263–272 (2017)
21. Wang, Y., Zhou, Q., Liu, J., et al.: LEDNet: a lightweight encoder-decoder network for real-time semantic segmentation (2019)
22. He, K., Zhang, X., Ren, S., et al.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision & Pattern Recognition. IEEE Computer Society (2016)
23. Chollet, F.: Xception: deep learning with depthwise separable convolutions. arXiv e-prints (2016)
24. Lee, Y., Hwang, J., Lee, S., et al.: An energy and GPU-computation efficient backbone network for real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2019)
25. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift (2015)

26. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: Proceedings of International Conference on Machine Learning (ICML), pp. 807–814 (2010)
27. Ma, N., Zhang, X., Huang, J., et al.: WeightNet: revisiting the design space of weight networks. arXiv preprint [arXiv:2007.11823](https://arxiv.org/abs/2007.11823) (2020)
28. Cordts, M., Omran, M., Ramos, S., et al.: The cityscapes dataset for semantic urban scene understanding (2016)
29. Brostow, G.J., Shotton, J., Fauqueur, J., Cipolla, R.: Segmentation and recognition using structure from motion point clouds. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008. LNCS, vol. 5302, pp. 44–57. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88682-2_5
30. Li, G., Yun, I., Kim, J., et al.: DABNet: depth-wise asymmetric bottleneck for real-time semantic segmentation (2019)
31. Yu, C., Wang, J., Peng, C., et al.: Learning a discriminative feature network for semantic segmentation. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (2018)
32. Chen, L.C., Papandreou, G., Kokkinos, I., et al.: DeepLab: semantic image segmentation with deep convolutional nets, Atrous convolution, and fully connected CRFs. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**(4), 834–848 (2018)
33. Wang, Y., Zhou, Q., Liu, J., et al.: LEDNet: a lightweight encoder-decoder network for real-time semantic segmentation. In: 2019 IEEE International Conference on Image Processing (ICIP), pp. 1860–1864. IEEE (2019)
34. Zhang, B., Li, W., Hui, Y., et al.: MFENet: multi-level feature enhancement network for real-time semantic segmentation. *Neurocomputing* (2020)
35. Zhuang, J., Yang, J., Gu, L., et al.: ShelfNet for fast semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (2019)



Parallel Reconstruction for High Resolution Multi-frame Solar Speckle Images

Lingxiao Zhu^(✉), Murong Jiang, Pengming Fu, and Wenhao Chui

School of Information Science and Engineering,
Yunnan University, Kunming, China

Abstract. The high resolution reconstruction of solar speckle image is an important research issue in the field of astronomical observation. Due to the influence of atmospheric turbulence, the astronomical images obtained by ground-based optical telescope will be blurred or degraded seriously, which needs to be reconstructed by image restoration method. Most of the existing regularization methods deal with a single frame image. The more image features are, the better the reconstruction effect is. However, the poor quality of the reconstructed image is caused by the insufficient features of a single solar speckle image. In this paper, we combined the complementary relationship between multi-frame speckle images to establish a multi-frame blind restoration model suitable for solar speckle image reconstruction, used genetic algorithm to select regularization parameters and realized multi-frame block reconstruction in parallel. The experimental results show that the proposed method can restore the solar speckle image well, and the reconstruction speed is fast, which can meet the requirements of astronomical observation.

Keywords: Solar speckle image · Regularization · Blind image restoration · Parallel computing

1 Introduction

Solar speckle image [1] reconstruction is one of the important research contents in astronomical image processing. In the solar high-resolution observation, affected by atmospheric turbulence, image acquired by ground-based optical telescope will be severely blurred or degraded, and high-resolution image reconstruction is needed. There are two kinds of solar speckle reconstruction techniques: one is frequency reconstruction algorithm, such as speckle interferometry [1, 2], speckle-masking, Knox-Thompson method; the other is to complete the reconstruction directly in spatial domain, such as simple displacement superposition and weighted displacement superposition. These reconstruction algorithms are based on atmospheric physics model. In the reconstruction process, more prior knowledge is required, such as atmospheric visibility, speckle interference function, and more image frames are

Supported by the Program for Innovative Research Team (in Science and Technology) in University of Yunnan Province (IRTSTYN).

required, which results in a large amount of calculation in the reconstruction process and fails to meet the real-time requirements of astronomical observation.

The solar speckle image contains more noise information, so blind restoration is needed by using blind restoration method in image processing. Blind restoration can solve the restored image and the unknown blur kernel at the same time. The number of unknowns is more than the number of equations, which is a serious ill posed problem. Regularization method is a classical method proposed by Tikhonov [3] in 1977 for solving ill-posed problem. It transforms ill-posed problem into posed problem by introducing regularization constraint of image smoothness. Tikhonov used Gaussian prior to fit the heavy tailed distribution of natural images, and constructed a regularization term about the image gradient. Although the noise can be suppressed, the effect of deblurring is not good and the details are lost seriously. Rudin [4] proposed the total variation regularization method, using Laplace prior to fit the gradient heavy tailed distribution of natural images. It can get better restoration effect, but it will produce certain ringing effect. Krishnan [5] proposed a normalized sparse prior of image to estimate the simple blur kernel. On this basis, Tang Meng [6] first carried out filtering pretreatment on the image to reduce the noise and highlight the edge, and then used the total variation regularization method to perform non-blind deconvolution. The calculation process is complex. M. Van Noort [7] applied blind deconvolution to the restoration of astronomical turbulence degradation image, and proposed a blind deconvolution algorithm for solar speckle image.

Due to the single feature of the solar speckle image, if the traditional single frame reconstruction algorithm is used, the resolution of the reconstructed image will be low and the restoration effect will be poor. The multi-frame reconstruction algorithm can make full use of the similar information between the image frames to complement each other, so that the reconstructed image can obtain more details and information, which is closer to the ideal target image.

In this paper, we establish a multi-frame blind restoration model suitable for solar speckle image by using the continuity feature between multi-frame images. The regularization parameters in the model are calculated by genetic algorithm, which is taken as the initial regularization parameter. After iterative calculation, the blur kernel and reconstructed image are obtained. The results show that the proposed method does not require much computation, and only a few frames can be used to restore the clear images that meet the needs of astronomical observation.

The structure of this paper is as follows: the second part describes in detail the establishment process and calculation method of multi-frame blind restoration model; the third part uses parallel algorithm to realize multi-frame block reconstruction; the fourth part summarizes the methods and shortcomings of this paper.

2 Calculation of Multi-frame Blind Restoration Model

2.1 Single Frame Blind Restoration Model

In the calculation of blind restoration of a single frame image, the most representative regularization method is the L2 regularization constraint proposed by Tikhonov in 1977. The model is:

$$\operatorname{argmin}_{f,h} \left(\|f * h - g\|^2 + \alpha \|h\|^2 + \beta B(f) \right) \quad (1)$$

where f and h represent the observed image and the blur kernel, respectively; g is the blurred image, $\|f * h - g\|^2$ is the data fidelity term. The smaller the value is, the closer the restored image is to the original clear image. Similarly, the smaller the regular term of the blur kernel is, the closer the estimated blur kernel is to the real kernel. $B(f)$ is the other information of the image, which is defined according to different prior knowledge. α , β is the regularization parameter, which play the role of balancing the regularization term and the data term [6].

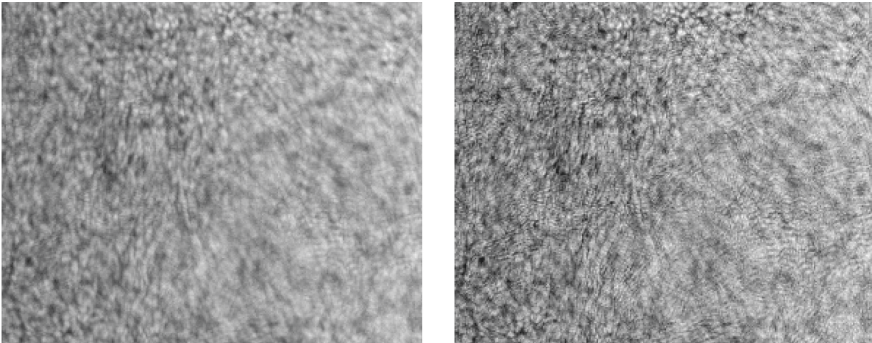
In the model (1) based on image gray scale and image gradient, $B(f)$ is defined as:

$$B(f) = \sigma P_0(f) + P_0(\nabla f) \quad (2)$$

$P_0(f) = \|f\|_0$ and $P_0(\nabla f) = \|\nabla f\|_0$ is the number of image f and gradient image ∇f nonzero element. Combining (1) and (2), the regularization model of blind image restoration can be expressed as:

$$\operatorname{argmin}_{f,h} \left\{ \|f * h - g\|^2 + \alpha \|h\|^2 + \beta \sigma P_0(f) + P_0(\nabla f) \right\} \quad (3)$$

The process of image restoration is essentially the deconvolution operation of the matrix. Due to the lack of detailed features contained in the single frame of the solar speckle image, the noise is amplified during the reconstruction operation, resulting in unsatisfactory reconstruction results, as shown in Fig. 1.



(a) Blurred image

(b) Reconstructed image by formula(3)

Fig. 1. Reconstruction result of single speckle image

2.2 Multi-frame Blind Restoration Model

In astronomical observation, the formation process of solar speckle image is linear or spatially invariant in the isoplanatic region [2], and the noise is mainly additive and independent of the observation target. The imaging model of multi-frame observation image is [8]:

$$g_i = f * h_i + n_i, i = 1, 2, 3, \dots, k \quad (4)$$

where f is the original image, h_i is the blur kernel of frame i , n_i is the noise in the image of frame i , and g_i is the blur image of frame i .

If we do not consider the existence of noise, we can get the relationship between every two frames of images.

$$g_i * h_j = g_j * h_i, i \neq j \quad (5)$$

However, in the actual astronomical observation, noise exists and is a mixture of multiple noises, so we should add the correction operator s to modify h_i . Since the relationship (5) still holds with s :

$$g_i * h_j * s = g_j * h_i * s, i \neq j \quad (6)$$

The multi-frame image degradation model can be written as:

$$\begin{cases} g_i = f * h_i + n_i, i = 1, 2, \dots, k \\ g_i * h_j * s = g_j * h_i * s, i \neq j \end{cases}$$

Base on (5), we order

$$A(\{h_i\}) = \sum_{m,n \in i, m \neq n} \|g_m * h_n - g_n * h_m\|^2 \quad (7)$$

It represents the difference of multi-frame image after restoration, which is taken as a new regularization constraint, and the regularization model of multi frame image reconstruction is obtained based on (3).

$$\operatorname{argmin}_{f, \{h_i\}} \left(\sum_i \|f * h_i - g_i\|^2 + \alpha A(\{h_i\}) + \beta(\sigma P_0(f) + P_0(\nabla f)) \right) \quad (8)$$

The solution of problem (8) can be divided into the following two sub-problems:

Sub-problem 1: solving blur kernel $\{h_i\}$

$$\operatorname{argmin}_{\{h_i\}} \left(\sum_i \|f * h_i - g_i\|^2 + \alpha A(\{h_i\}) \right) \quad (9)$$

Sub-problem 2: solving the original image f

$$\operatorname{argmin}_f \left(\sum_i \|f * h_i - g_i\|^2 + \beta(\sigma P_0(f) + P_0(\nabla f)) \right) \quad (10)$$

The following is the derivation of the sub-problem solving process.

1) Solution of problem 1.

In Eq. (9), $A(\{h_i\})$ will be disturbed by noise. According to Eq. (6), it should be rewritten as:

$$A(\{h_i\}) = \sum_{m,n \in i, m \neq n} \|g_m * h_n * s - g_n * h_m * s\|^2$$

Because s is unknown, problem 1 turns into solving $\{h_i\}$ and s by using.

$$\operatorname{argmin}_{\{h_i\}, s} \left(\sum_i \|f * h_i - g_i\|^2 + \alpha A(\{h_i\}) \right)$$

First of all, we get the initial estimate of h according to $h = \operatorname{argmin}_h A(\{h_i\})$. The degradation model of blur kernel can be regarded as:

$$h^0 = h * s + \epsilon \quad (11)$$

The regularization method can also be used to solve Eq. (11). Add regularized constraint items to build the target optimization problem:

$$\operatorname{argmin}_{s, \{h_i\}} \left(\sum_i \|h_i * s - h_i^0\|^2 + \delta\varphi(s) + \delta\varphi(\{h_i\}) + \gamma\omega_p(\{h_i\}) \right) \quad (12)$$

where $\varphi(\cdot)$ represent positive constraint:

$$\varphi(h) = \sum_X p(h(X)), p(t) = \begin{cases} t, & \text{if } t \geq 0 \\ +\infty, & \text{otherwise} \end{cases}$$

We decompose Eq. (11) to solve $\{h_i\}$:

$$\operatorname{argmin}_{\{h_i\}} \left(\sum_i \|h_i * s - h_i^0\|^2 + \delta\varphi(\{h_i\}) + \gamma\omega_p(\{h_i\}) \right)$$

and s :

$$\operatorname{argmin}_s \left(\sum_i \|h_i * s - h_i^0\|^2 + \delta\varphi(s) \right)$$

1) Solution of problem 2.

We use the result of problem 1 to solve the original image f :

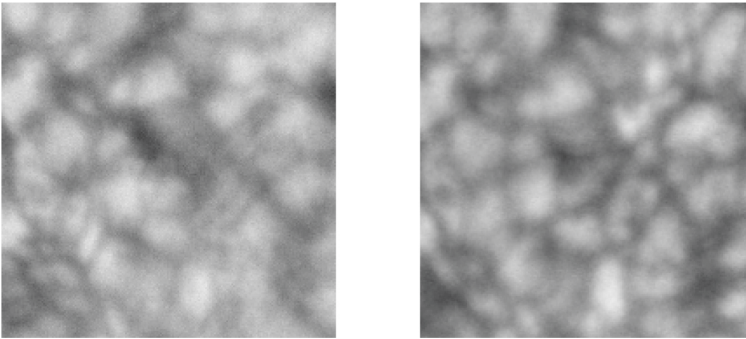
$$\operatorname{argmin}_f \left(\sum_i \|f * \hat{h}_i - g_i\|^2 + \beta(\sigma P_0(f) + P_0(\nabla f)) \right) \quad (13)$$

By means of Fourier transform, the solution of Eq. (13) is

$$f = F^{-1} \left(\frac{\sum \overline{F(\hat{h}_i)} F(g_i)}{\sum \overline{F(\hat{h}_i)} F(\hat{h}_i) + \beta \overline{F(P)} F(P)} \right)$$

2.3 Calculation of Regularization Parameters

In the Multi-frame regularization model (8)–(12), the regularization parameters are $\alpha, \beta, \gamma, \delta, \sigma$. The selection of these parameters not only affects the restoration result, but also affects the convergence of the iterative calculation process. The wrong regularization parameters will lead to the worse quality of reconstructed image. The traditional regularization parameter calculation method is not suitable. We use Genetic algorithm to solve the optimal regularization parameter. For the same image, regularization parameters and experimental parameters selected by genetic algorithm were used for reconstruction, and the results were shown in Fig. 2. It can be explained that the reconstruction effect of regularization parameters selected by genetic algorithm is better.



(a) Experimental parameters: $\alpha = 10$,
 $\beta = 1, \gamma = 20, \delta = 7, \sigma = 10$

(b) Parameters selected by GA: $\alpha = 17$,
 $\beta = 1.21, \gamma = 27.31, \delta = 13.32, \sigma = 9.65$

Fig. 2. Reconstruction results with different regularization parameters

3 Parallel Implementation of Multi-frame Block Reconstruction

The multi-frame regularized reconstruction model is effective in the isoplanatic region, and the multi-frame image should be segmented and reconstructed according to the size of the isoplanatic region. In addition, the image size of the speckle image and the number of frames is large. These will lead to a long time required for the algorithm reconstruction in this paper. Therefore, it is necessary to redesign the optimization algorithm and introduce parallel processing [9].

Parallel algorithm is a method to solve a problem by using multiple processors. The execution process is to divide a given problem into several sub-problems which are as independent as possible, and then use multiple processors to calculate simultaneously, and finally the original problem is solved. The design strategy adopted in this paper is to make full use of the parallelism of the existing serial algorithm and direct parallelization of the serial algorithm [10]. The serial reconstruction process is divided into three steps: 1. Sub-block division 2. Sub-block reconstruction 3. Sub-block stitching. The running flow chart of the parallel program is as follows (Fig. 3):

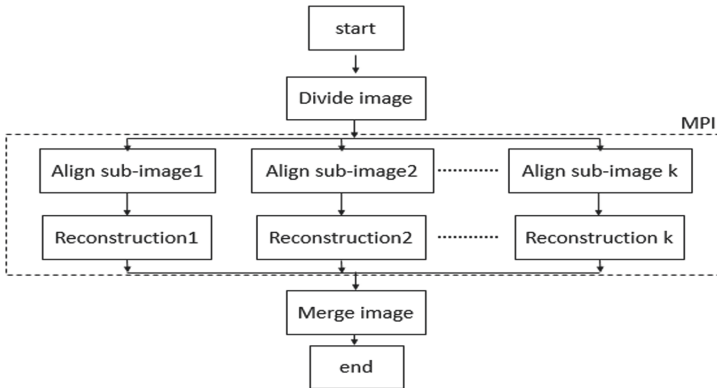


Fig. 3. Program chart.

MPI parallel mode is adopted to allocate tasks to each thread, and we take the sub-block size of 512 pixel \times 512 pixel as an example. The reconstructed image is divided into 16 blocks, as shown in Fig. 4. If the number of thread can be divided exactly by tasks, a task can be assigned to each thread integer, but when the number of thread cannot be divided exactly by number of jobs, for each thread in task allocation, scheduling policy must be considered in order to realize load balance, otherwise there will be a thread out tasks that other threads only computing tasks little or no computing tasks. It could cause the waste of computing resources and the acceleration effect is not ideal. To solve this problem, we adopt the strategy: treat all threads as a pool, assign a task to each thread, and redistribute a new task until all tasks are completed.

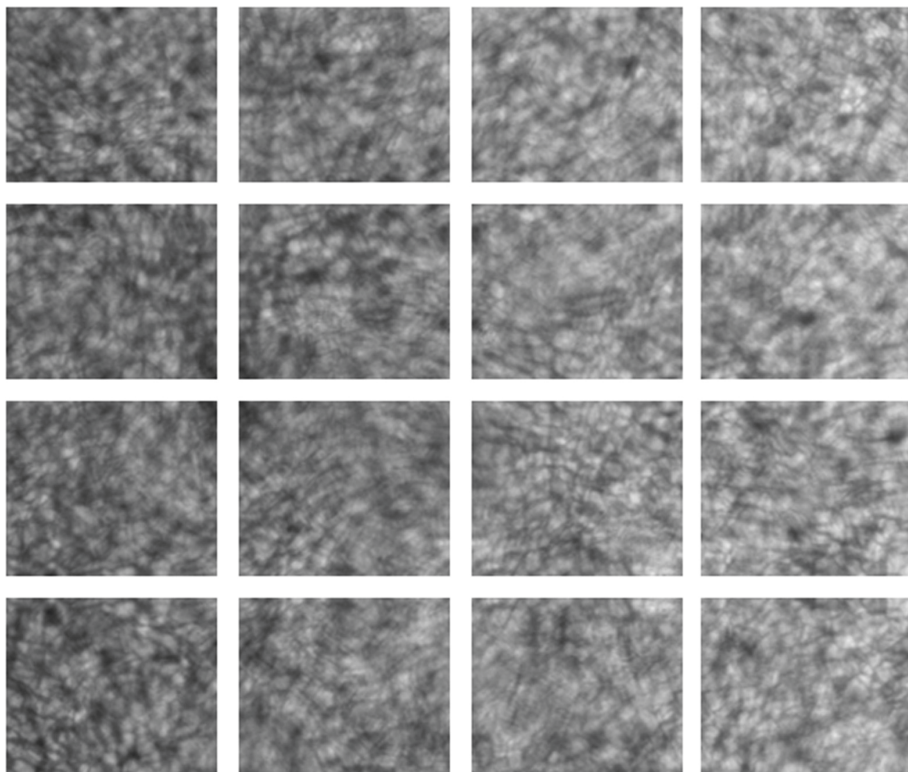


Fig. 4. Schematic diagram of sub-block division

Using MATLAB platform, the program runs on a Dell workstation equipped with Intel I7-8700K and 64 GB memory. We adopt 200 frames of 1912×2360 solar observation images obtained by Yunnan Observatory at 13:30 on March 3, 2019, and use the frame selection algorithm to select 10 frames of images, and then divide sub-block to reconstruct a target image in parallel. Figure 5 shows the result of Multi-frame block reconstruction.

We test the runtime and speedup of your program using Intel's vtuneamplifier software. As shown in Fig. 6, the circle indicates that the size of the sub block is 256×256 , triangle indicates the size of sub block 512×512 . With the same size, as the number of CPU cores increases, the running time decreases and the relative acceleration ratio increases. When processing 1912×2360 image with 6 cores, the maximum speedup of the program is 7.21 when the sub-block is selected as 512×512 . In Fig. 5 (c), when the number of cores is 4, the sub-block size is 512×512 , and the number of sub-blocks is 4 times the number of cores. So the program can achieve a more ideal speedup than 6.17 than the speedup of 6 cores is higher than 6.03.

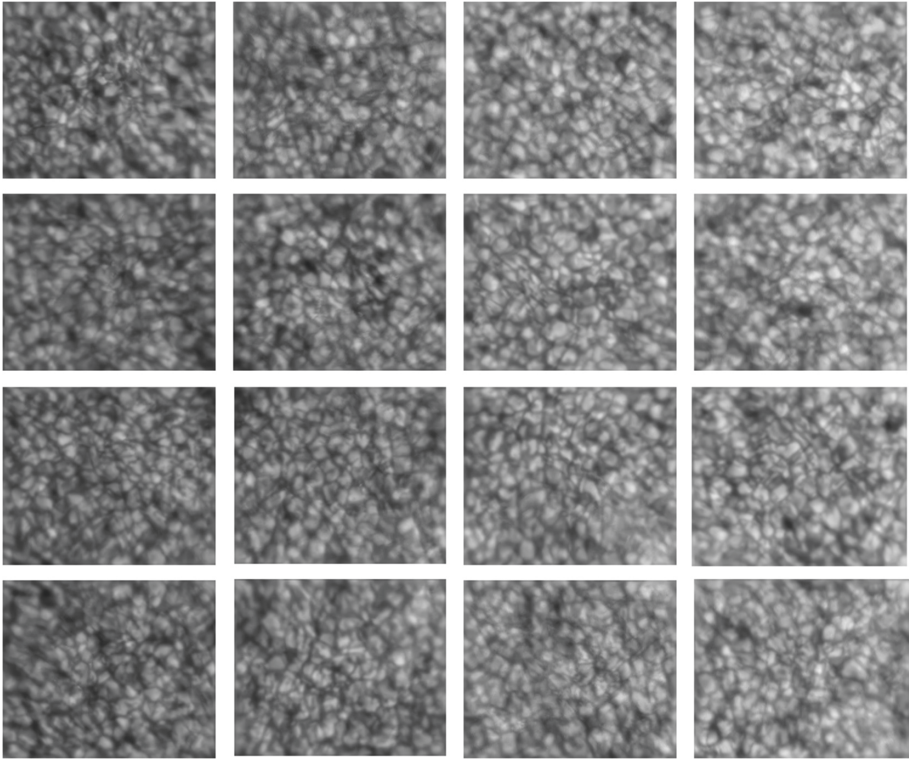


Fig. 5. The result of Multi-frame block reconstruction

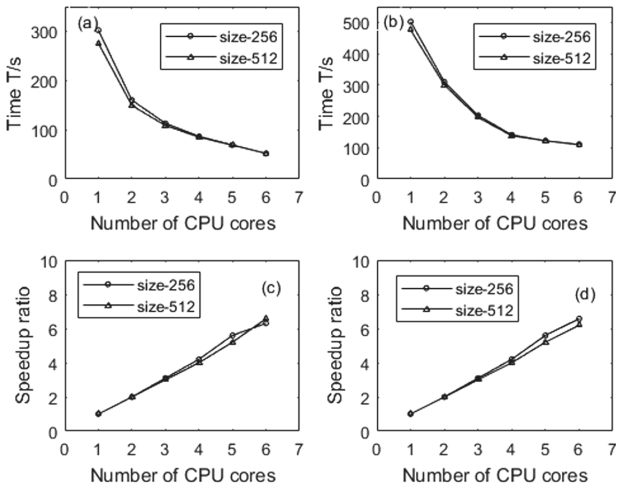
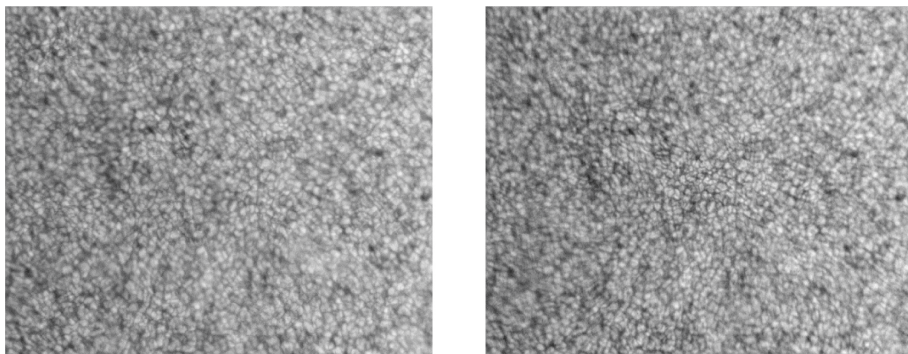


Fig. 6. Elapsed time and speedup ratio with different numbers of CPU cores



(a) Reconstructed result of figure 1(a)

(b) Reconstructed result of other data

Fig. 7. Reconstruction results of two groups of data on the same day

We reconstruct another set of images acquired on the same day, as shown in Fig. 7. The experimental results show that the edge contour of multi-frame block reconstruction is clear and the detailed contours of rice grains are visible. Compared with the single-frame blind restoration, the result is improved in vision, the details are enhanced, and the local details are more intuitive, which can better restore the sun speckle image. The parallel implementation of block reconstruction greatly improves the speed of multi frame image reconstruction, and can meet the real-time needs of astronomical observation.

4 Conclusions



In this paper, aiming at the blind restoration of the sun speckle image, we study the single-frame regularization blind restoration algorithm, and on this basis, use the relationship between the images to establish a multi-frame image reconstruction model, introduce genetic algorithm to calculate the regularization parameters, and parallelize the reconstruction process. Experiments show that the method in this paper uses the complementary information contained in multiple frames of images, while achieving image reconstruction, while maintaining edges and suppressing noise, it has good noise resistance, effectively improves the quality of solar speckle image reconstruction, and has high time efficiency. Due to the limitation of experimental conditions, the time efficiency of this method can be further improved.

References

1. Huo, Z., Zhou, J.: Method of astronomical image reconstruction from speckle pattern. *Prog. Astron.* **01**, 74–94 (2010). [https://doi.org/10.3969/j.issn.1000-8349.2010.01.005\(inChinese\)](https://doi.org/10.3969/j.issn.1000-8349.2010.01.005(inChinese))
2. Xiang, Y., Liu, Z., Jin, Z., et al.: High resolution solar image reconstruction method. *Prog. Astron.* **1**, 94–110 (2016). <https://doi.org/10.3969/j.issn.1000-8349.2016.01.06>. (in Chinese)
3. Van Noort, M., Der Voort, L.R.V., Löfdahl, M.G.: Solar image restoration by use of multi-frame blind de-convolution with multiple objects and phase diversity. *Solar Phys.* **228**(1) 191–215 (2005). <https://doi.org/10.1007/s11207-005-5782-z>
4. Zhu, X., Sroubek, F., Milanfar, P.: Deconvolving PSFs for a better motion deblurring using multiple images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) *Computer Vision – ECCV 2012*. LNCS, vol. 7576, no. 1, pp. 636–647. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33715-4_46
5. Yang, Y., Tao, Z., Liu, H.: Parallel computation methods for static security-constrained optimal power flow of power system. *Electr. Power Autom. Equip.* **039**(001), 99–105 (2019)
6. Tang, M., Peng, G., Zheng, H.: Image blind deblurring based on regularization method. *Comput. Appl. Res.* **31**(2), 596–599611 (2014). (in Chinese)
7. Afonso, M.V., Bioucas-Dias, J.M., Figueiredo, M.A.T.: Fast image recovery using variable splitting and constrained optimization. *IEEE Trans. Image Process.* **19**(9), 2345–2356 (2010)
8. Libo, Z.: Research on solar high resolution image reconstruction. University of Chinese Academy of Sciences, Beijing (2015)
9. He, B., Chen, N., Lin, D., et al.: An efficient parallel computing method for the steady-state analysis of low-frequency electromagnetics using an anti-periodic condition. *IEEE Trans. Magn.* **PP**(99), 1–1 (2020). <https://doi.org/10.1109/TMAG.2019.2957695>
10. Lin, C., Jian, L., Jun, Z., et al.: Multithreading method to perform the parallel image registration. In: *International Conference on Computational Intelligence & Software Engineering*. IEEE (2009)



Analysing and Forecasting Electricity Demand and Price Using Deep Learning Model During the COVID-19 Pandemic

Israt Fatema^(✉) , Xiaoying Kong, and Gengfa Fang 

University of Technology Sydney, Ultimo, NSW 2007, Australia
israt.fatema@student.uts.edu.au

Abstract. The smart city integrating the smart grid as an integral part of it to guarantee the ever-increasing electricity demand. After the recent outbreak of the COVID-19 pandemic, the socioeconomic severances affecting total levels of electricity demand, price, and usage trends. These unanticipated changes introducing new uncertainties in short-term demand forecasting since its result depends on the recent usage as an input variable. Addressing this challenging situation, this paper proposes an electricity demand and price forecast model based on the LSTM Deep Learning method considering the recent demand trends. Real electricity market data from the Australian Energy Market Operator (AEMO) is used to validate the effectiveness of the proposed model and elaborated with two scenarios to get a wider context of the pandemic impact. Exploratory data analyses results show hourly electricity demand and price reductions throughout the pandemic weeks, especially during peak hours of 8 am–12 noon and 6 pm–10 pm. Electricity demand and price has been dropped by 3% and 42% respectively on average. However, overall usage patterns have not changed significantly compared to the same period last year. The predictive accuracy of the proposed model is quite effective with an acceptably smaller error despite trend change phenomena triggered by the pandemic. The model performance is comprehensively compared with a few conventional forecast methods, Support Vector Machine (SVM) and Regression Tree (RT), and as a result, the performance indices RMSE and MAE have been improved using the proposed LSTM model.

Keywords: COVID-19 · Pandemic · Electricity demand and price forecast · LSTM · Smart grid · Smart city

1 Introduction

Smart City can efficiently address the challenges of a growing population to manage their essential activities, such as energy, transport, health, and homes. A reliable and sustainable Smart Grid (SG) is essential for affordable electricity to smart city's consumers. However, the pandemic of COVID-19 affects almost all aspects of the community and eventually the concept of a smart grid. Due to the present circumstances of COVID-19, mass people are working from home, and the forced closure of industries and other commercial activities significantly slacked down daily activities in comparison

with the non-pandemic ones. The socioeconomic severances naturally affecting on total levels of electricity consumption, demand, price, and usage trends worldwide.

This changed working condition eventually reflected in electricity grid planning, demand scheduling, renewable source integrating, and spot pricing. Electricity demand and price forecasting have important roles in the economy, which is frequently used in business planning, policymaking, and market setting. Smart grids are depended on a forecasting model that is mostly designed and validated on historical data. However, there are no historical time-series smart grid data presents comparable to the COVID-19 pandemic period. Consequently, the short-term forecasting algorithm's performance is affected by the aforementioned uncertainty. Therefore it is imperative to improve the forecasting accuracy in terms of the possible error reduction.

Traditional forecasting methods, such as moving average (MA) and trend analysis, get complicated and limited if used in large time-series data set [1]. These methods are challenging to accurately measured and represented with detail dynamic operations occurred because of the recent trend shift. Researchers have applied deep-learning methods, such as Artificial Neural Network (ANN), to improve the models' prediction accuracy by reducing errors and modelling complex patterns. In the context of such uncertain times of pandemics, this paper aims to contribute and address the problems of implementing accurate and reliable demand and price forecast algorithm. First, an initial exploratory analysis of electricity demand and price time-series data is performed to compare the diurnal variation during the pandemic and non-pandemic periods. Statistical analysis is applied to evaluate the inconsistency and uncertainty of the forecasting problem. Secondly, a single comprehensive model of the LSTM based sequence-to-sequence network is proposed to forecast electricity demand and price. This model can effectively learn variable temporal correlation in the input sequence, which is commonly used for language translation [2]. Similar temporal correlations are also present in the electricity demand and price pattern.

The arrangement of this paper is as follows: related work is given in Sect. 2. The theoretical background of this work is discussed in Sect. 3. Section 4 describes the dataset used for this study. An empirical analysis of data is in Sect. 5. The modeling results and discussions are presented in Sect. 6. Section 7 concludes the article.

2 Related Work

Deep learning has been increasingly involved in the forecasting methods [3] and is being applied effectively in various time series issues, such as language modelling [4], speech recognition [5], stock market prediction [6], and flood forecasting [7]. In [8], the researchers used a recurrent neural network (RNN) as an important approach for time-series forecasting. However, RNN shows inadequacy in learning long term dependencies and relies on fixed-term to learn time-series sequence computation [9]. LSTMs, are a special type of RNN, could learn long-term dependencies by remembering and collecting information of time-series [10]. Few other approaches were also implemented to enhance electricity demand or price forecasting performance, such as feature selection and genetic algorithm to optimize a LSTM model [11], support vector regression (SVR), stacked auto-encoders (SAEs) with the extreme learning machine

(ELM) [12], and stacked de-noising auto-encoders with SVR [13]. It is observable that the ANN-RNN based models are very common for related fields of electricity.

The aforementioned methods could give good results, yet its algorithms are complicated and difficult. This paper, therefore, proposes a single comprehensive model of the LSTM based sequence-to-sequence network to forecast electricity demand and price. Though such a model has recently been used in energy and weather forecasts, its application has not been used widely in terms of electricity demand and price forecast. In [14], LSTM was compared to the sequence-to-sequence network and in [15], sequence-to-sequence RNN was compared to standard RNN. In both studies, the sequence-to-sequence network provided better results than other models. A sequence-to-sequence RNN was developed with an attention mechanism for the electric load forecast in recent research [15], and a similar sample generation method was designed. In [16], A LSTM-based short-term load forecast model with two mechanisms is built. Their approach is similar to the language translation model in [2], where one LSTM is used to encode the input sequence into a fixed vector, and then need separate LSTM to decode the vector to a sequence of outputs with an attention mechanism to learn weight. The attention mechanism learns to weight the input features variable conditioned on the previous input(s) rather than fixed weighting features [2]. Alternatively, this research uses a simplified approach using a single LSTM that encodes and decodes both on the basis of the given inputs. This allows the LSTM to share weights between encoding and decoding. Therefore, no complicated attention mechanism is required for this straightforward sequence-to-sequence model.

3 Theoretical Background

3.1 Long Short Term Memory (LSTM) Network

The LSTM's key objective is to prevent the issue of vanishing gradient which occurs while training of backpropagation neural network (NN), and thus limiting the model's ability to learn long-term temporal correlations [10]. All RNN follow the structure of a chain of recurring modules of NN. For regular RNNs, this recurring module will have a very simple structure, like a single *tanh* layer. The structure of LSTM includes additional three main gate structures: forget gate (f_t), input gate (i_t), and output gate (o_t). Based on the LSTM unit defined in [17], for an input x_t at time step t , the LSTM calculate a hidden state h_t and memory cell state C_t to encode all the observed state by the cell till time t . The LSTM network computes a mapping from an input sequence $X = (x_1, x_2, \dots, x_n)$ to an output sequence $Y = (y_1, y_2, \dots, y_m)$. The LSTM cell computation at time t , for an input x_t :

$$f_t = \delta(W_f x_t + U_f h_{t-1} + b_f) \quad (1)$$

$$i_t = \delta(W_i x_t + U_i h_{t-1} + b_i) \quad (2)$$

$$g_t = \tanh(W_g x_t + U_g h_{t-1} + b_g) \tag{3}$$

$$C_t = i_t * g_t + f_t * C_{t-1} \tag{4}$$

$$o_t = \delta(W_o x_t + U_o h_{t-1} + b_o) \tag{5}$$

$$h_t = o_t * \tanh C_t \tag{6}$$

where δ and \tanh are the activation function, W and U are the weight of forget gate, and b is the bias vector, C_{t-1} and C_t are the cell states at time $t-1$ and t .

3.2 Sequence-To-Sequence Network

To forecast the values of future time steps of a sequence, a sequence-to-sequence regression LSTM network can be trained. A typical sequence-to-sequence model consists of two phases, an encoder, and a decoder. It can take input sequence X (encoder) of variable length and change that in a fixed-length vector, which is then used as the input sequences for the next time step [2]. Hereby an output sequence Y (decoder) of n length is generated. In this case, at each time step of the input sequence, the LSTM network learns to forecast the value of the next n time steps. Therefore, during encoding, with input sequence X , the LSTM computes a sequence of hidden states (h_1, h_2, \dots, h_n) . During decoding it defines a distribution over the output sequence Y given the input sequence X as $p(Y|X)$ is:

$$p(y_1, \dots, y_m | x_1, \dots, x_n) = \prod_{t=1}^m p(y_t | v, y_1, \dots, y_{t-1}) \tag{7}$$

where v is fixed dimensional vector representation of the X given by the last hidden state based on the recursion of the LSTM Eqs. (1)–(6), and the distribution of $p(y_t | v, y_1, \dots, y_{t-1})$ is given by a softmax function (Eq. 8) to create a probability vector that helps determine the final output.

$$p(y) = \text{softmax}(w^t h_t) \tag{8}$$

where the *softmax* activation function calculates the probability of each recurrent weight using the state hidden (h_t) at current time step with each weight (w^t) .

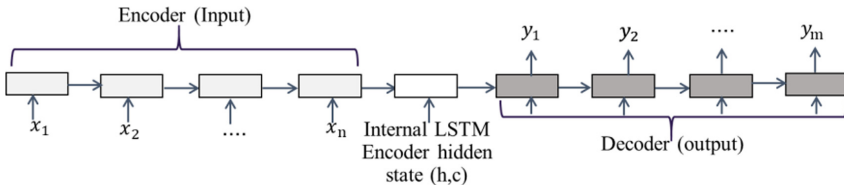


Fig. 1. The structure of Sequence-to-Sequence Network, (h,c) represents intermediate vector

The proposed method, shown in Fig. 1, relies on a single LSTM for both the encoding and decoding phases. Thus parameter is shared within the encoding and

decoding phases. The proposed model uses two LSTMs layers, the hidden state (h_t) from the first LSTM layer is given as the input (x_t) to the second LSTM. The first LSTM layer is used to create the input sequence for time series data and the second layer is used to create the prediction by the next output sequence. Forecasting model performance is measured by Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and correlation coefficient value (R^2). A lower RMSE and MAE indicates better result, which measures the difference between the actual and forecast values. The R^2 value is between 0 and 1 (0 means no correlation and 1 means no error), determines the correlation between actual and predicted values. Figure 2 shows the step-by-step flowchart of this study that includes two steps: an analysis module and a forecasting module.

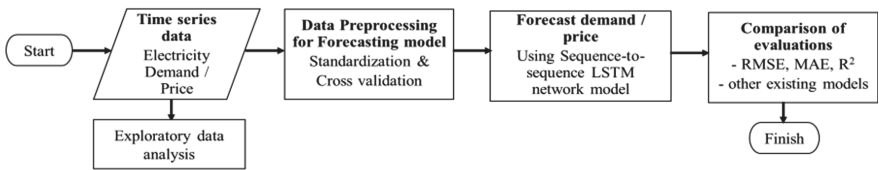


Fig. 2. Flowchart of the proposed forecasting method

3.3 Forecasting Model Description

The proposed model used Dropout as a regularization methodology for fully connected neural network layers to avoid over-fitting, and improving accuracy on testing data [18]. All elements of an output layer are stored with probability p , alternatively set to 0 with probability $(1 - p)$. Equation (9) shows in this case drop unit or not [18].

$$P(n) = \begin{cases} 1 - p & \text{for } n = 0 \\ p & \text{for } n = 1 \end{cases} \quad (9)$$

The simulation result's precision is improved by adjusting and setting the model's appropriate variables to produce the desired output. The model parameter was set in a total of 36 settings to achieve best result. The initial learning rate of the experiment was set to 0.005, the number of hidden units was set to 200/100/50, set the number of iterations to 300/250/200, Dropout ($p = 0.5$) and Adam (Adaptive Moment Estimation) optimizer algorithm is used for optimization to update network weights.

4 Dataset and Data Preprocessing

It is not immensely challenging to train a forecasting algorithm using historical data. However, social trends changed quite quickly due to the COVID-19 pandemic effect, which may cause the models built using historical data ineffective and incorrect. Since there are only months of data available for model training and testing processes and

small amounts of data generally decrease model accuracy. To address this issue, AEMOs' open dataset is used, which contains accumulated daily electricity demand (30 min MW) and price (30 min/MWh) sampling rate [19]. This study used the range of data for the forecasting model is from January 2019 to August 2020. To analyse the impact on electricity demand and price profiles during COVID-19 and to evaluate the forecasting model, this study focuses on data from two states in Australia: New South Wales (NSW) and Victoria (VIC) as two different scenarios. The reasons for selecting these states are: (1) NSW is the highest populated state and the restriction set to ease due to reduced infection rate, (2) VIC has the highest number of COVID 19 cases compared to other states and is experiencing a second wave, not currently being seen in other states. Consequently, VIC is under strict restrictions.

To enhance model efficiency, the cross-validation technique is used for the assessment of the forecasting model [20]. Cross-validation stages include the splitting of the data set into training, and test data for unbiased performance comparison [20]. The data split of this research is 90% training and tests on the remaining 10%. The original data is pre-processed by standardization to improve the model prediction accuracy and to eliminate the training from deviating [21]. Standardize the data to have zero mean and unit variance to make it more effective:

$$X_{stand} = \frac{X - \mu}{\sigma} \quad (10)$$

Here, the standardized variable is X_{stand} which is equal to the original variable (x), minus its mean (μ), divided by its standard deviation (σ).

5 Impact on Electricity Demand and Price During COVID-19

The initial exploratory analysis of electricity demand and price data can be useful to better understand the dynamic changes and identify trends and patterns to the energy sector due to COVID-19. Due to the influence of several factors, such as temperature, day of the week, and variation of renewable sources, it is not easy and straightforward to determine the actual impact of the pandemic on the electricity demand and price.

In Australia, the nationwide restrictions, which started in mid-March to control the spread of COVID-19, also resulted in decreased demand for electricity and price. Figure 3(a-d) follows four years (2017–2020) data of two seasonal patterns, autumn (March-May) and winter (June-August) in NSW and VIC to show a comparison of change. As shown in Fig. 3, the NSW COVID-19 related demand decrease was highest compare to other year's similar period. Regardless, demand reductions were declining from the end of May due to the winter season and easing restriction. However, VIC's COVID-19 demand reduction was steady. For both scenarios, residential demand increased due to shut-down and working from home setup, and commercial demand was reduced due to limited business activities. Cold temperatures have resulted in an average increased in residential demand in both states (Fig. 4) since it is more related to weather than other industries. Figure 3 shows a drastic price reduction in both states during the COVID-19 period compare to the previous year's similar time. During the

pandemic, both states recorded its lowest average price since 2016 [19]. The pandemic has contributed to a major price drop in the international and local markets for crude oil, gas, and thermal coal [19]. This is a key factor of reduced spot price along with other factors including reduced demand and increased amount of renewable generation.

Figure 4(a–b) shows the changing pattern of daily average electricity demand, price, temperature, and renewable and non-renewable contribution to demand in NSW and VIC, during the pandemic period from 1st March to 31st August 2020 (data adopted from [22]). The figures suggest that it is difficult to capture the unexpected shifts in social behavior and pattern of work during the pandemic period since electricity generation (renewable and non-renewable), demand, price, and temperature change are not linear and not always following any pattern. Generally, the weather and different seasons influence the demand and price of electricity. Throughout the pandemic period, electricity demand shows a predictable trend and rises at a consistent rate to keep state with the temperature, unlike the electricity price that changes very randomly with a few sudden price peaks and intermittently following any pattern. Besides, increased renewable sources that include increased rainfall, wind, and solar production, have contributed to lower prices for both of the scenarios.

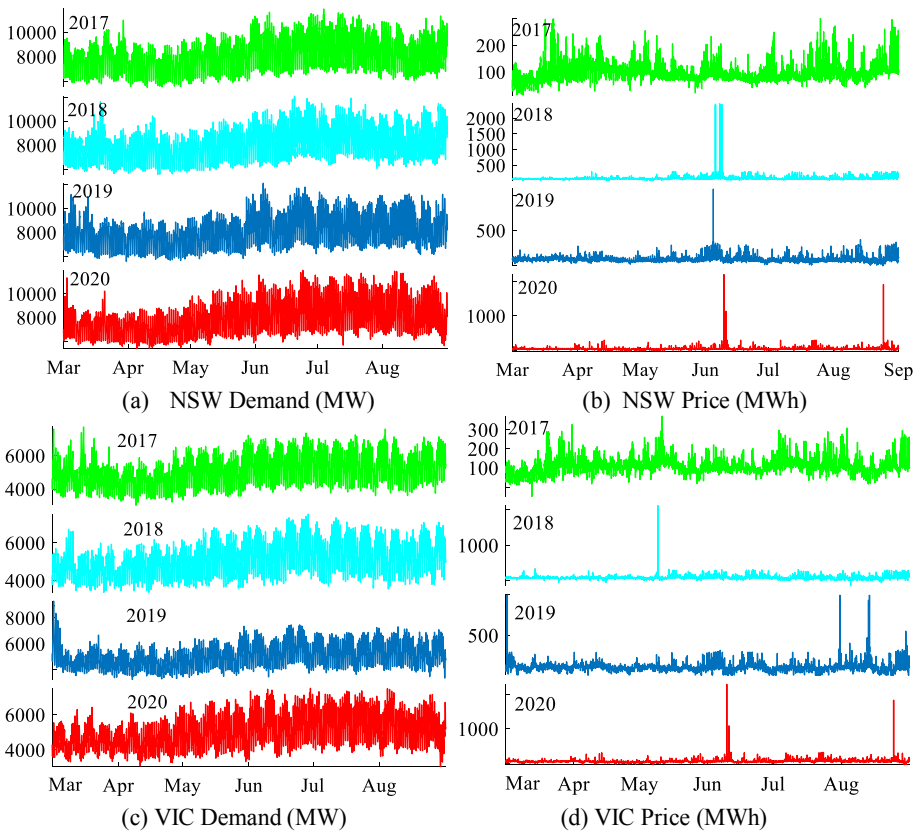


Fig. 3. Comparison of changing demand and price patterns for both scenarios from March–August in 2017–2020.

To analyse the daily real usage scenario during the pandemic period, demand/price data was grouped into weekdays and weekends, with the calculation of the average weekly time of use and compared to the same period of 2019. Hourly electricity demand and price show a clear reduction during the pandemic in NSW and VIC, especially during peak hours of 8 am–12 noon and 6 pm–10 pm. During the non-pandemic time, electricity price on weekdays is much higher than the weekend price, especially around peak hours in both scenarios. Compared with the price curve during the pandemic period, weekday and weekend are following a similar trend and overall price decreased in both peak and off-peak hours (Fig. 5).

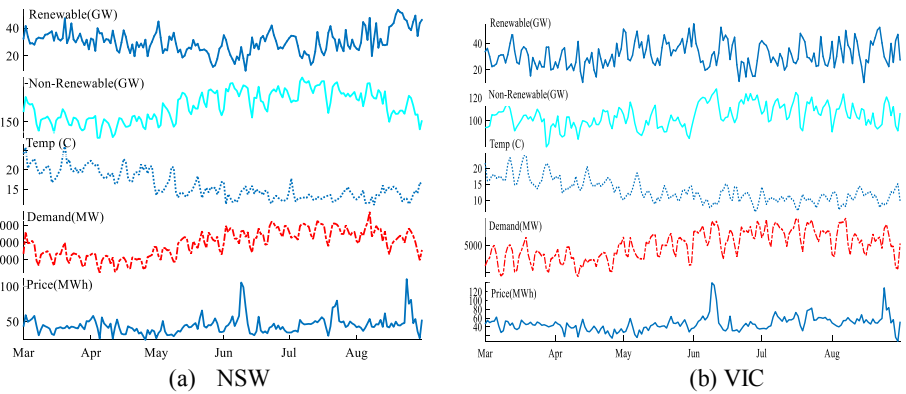


Fig. 4. The changing pattern of daily average electricity demand, price, temperature, and renewable and non-renewable contribution in (a) NSW and (b) VIC during the COVID-19 period.

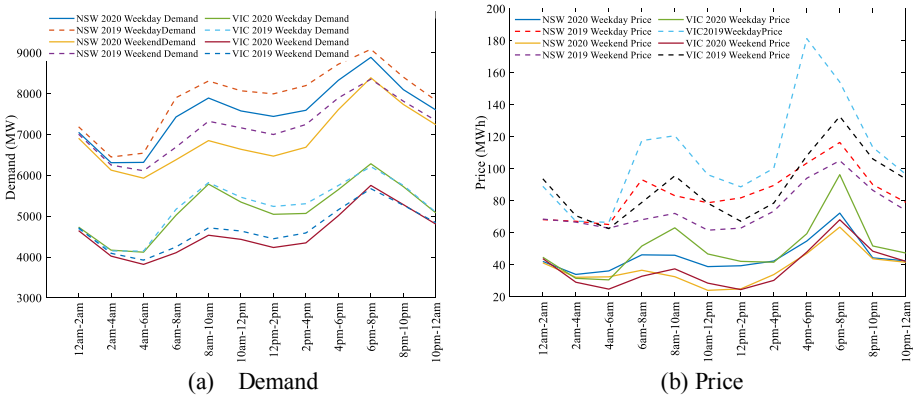


Fig. 5. Weekly electricity time of use during COVID-19 and the same period of 2019 in (a) NSW and VIC demand (b) NSW and VIC Price

Notably, overall demand and price patterns have not changed significantly compared to the same period last year (Fig. 5). This indicates people working from home are continuing regular activities and consuming electricity that remained usual morning and evening peak time trends during the COVID-19 pandemic. Statistical analysis is also performed to compare the changes. A Variation Index (VI) is defined which presents the average reduction of demand/price to show the diurnal variation during the COVID-19 period (2020) compared to a benchmark period (2019) as follows:

$$VI_{it} = \frac{\sum_{i=1}^n (D_{ct} - D_{ot}) \times 100}{n\bar{D}} \quad (11)$$

where VI_{it} is the index value of i for time t , D_{ct} is the current demand/price for time t (weekday and weekend), D_{ot} is the demand/price for the same time of a previous benchmark period, n is the number of recorded demand/price, and \bar{D} is the average demand/price during the previous benchmark period.

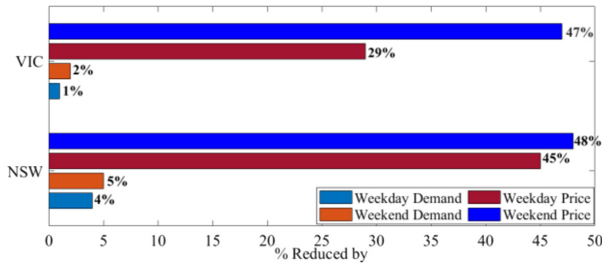


Fig. 6. Variation Index (VI) between pandemic and non-pandemic period for NSW and VIC

Figure 6 shows the VI for both of the states and the results show that NSW weekday electricity demand and price were decreased by 4% and 45% respectively on average, and on weekends it was 5% and 48% respectively during the strict restriction period (March–May). Compare to NSW, VICs’ demand reduction was lower on weekdays (average 1%), due to cooler weather. The weekend price was significantly dropped by 47% on average, whereas weekday prices reduced by only 29% on average.

6 Forecasting Model Results and Discussion

In this section, the proposed model forecasts the aforementioned two different scenarios (NSW and VIC) to get a wider context of the COVID-19 pandemic impact on electricity demand and price. The model has been trained and tested 9 times on independent data with different forecasting granularities (Sect. 3.3). The simulation results explain the performance of the proposed model in addition to the RMSE and MAE errors. Table 1 summarizes the best prediction results from 36 test results of two different scenarios.

Figure 7 shows that the forecasted and the observed values mostly conform to each other. The blue line in each figure represents the observed values and the red line implies the forecasted values. The forecast values quite closely match with the observed values of electricity demand data. However, compared to the demand forecast, spot pricing is not as accurate as expected in Fig. 7(b) and (d). Figure 8 displays the R^2 values for electricity demand and price for both scenarios. Whereas, the R^2 of the electricity price for NSW and VIC are .78 and .88 respectively, between actual and predicted values. This indicates a moderate correlation and similarity with Sect. 5 analysis that price VI was high and indistinct compare to demand during the pandemic period.

Table 1. Comparison of electricity demand and price forecasting errors and test result (summary).

		RMSE	MAE	Observed		Forecasted		Relative error	
				High	Low	High	Low	High	Low
NSW	Demand	58.21	40.51	11908.24	5630.73	11890.48	5567.92	0.17	0.62
	Price	29.67	7.36	1908.02	-12.84	1851.46	-20.35	0.56	0.07
VIC	Demand	41.44	30.69	7478.68	3054.77	7542.93	3058.37	0.60	0.036
	Price	33.46	10.76	1837.24	-48.72	1784.79	-42.45	0.52	0.06

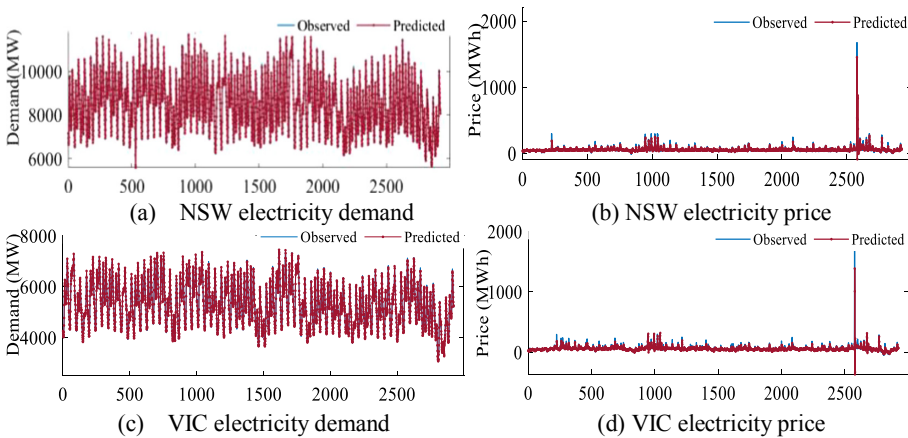


Fig. 7. Electricity demand and price forecast based on the model output with the assumptions of the first scenario (a)-(b) NSW and second scenario (c)-(d) VIC. (Color figure online)

To evaluate the reliability of the proposed LSTM model, the highest and lowest observed values from the test dataset are compared with the corresponding forecasted values. The forecasted highest and lowest electricity demand and price appear at the same time as the observed one in both scenarios. Both the highest records for demand

and price for two scenarios were on a weekday and the lowest records were on weekends. The figures (relative error) in Table 1 show that in all forecast cases, the first scenario has a tendency to provide a better performance in demand values prediction and the second scenario is showing better result for price values (Figs. 7 and 8).

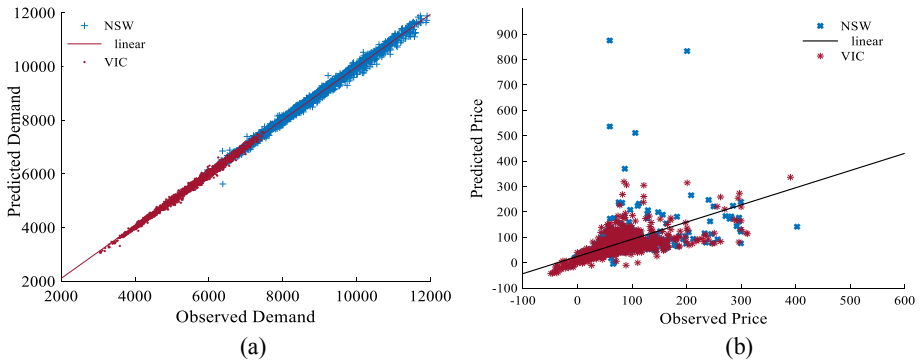


Fig. 8. Scatter plot to display coefficient correlation for electricity demand (a) and price (b) forecasting for both scenarios. (Color figure online)

To validate the performance of the proposed LSTM network, two other widely used conventional forecast methods, Support Vector Machine (SVM) and Regression Tree (RT), are also performed using the same dataset. Table 2 outlines the best possible results for each model. The findings indicate that the proposed model achieves the best results and compared to SVM and RT, the RMSE index of the proposed model has been averagely improved by 37% and 60% respectively in the case of electricity demand; similarly, the MAE has been improved by 39% and 47% respectively. In the case of electricity price, the performance error RMSE shows an improvement of 19% and 22% compared with SVM and RT respectively; similarly, MAE has been improved by 20% and 32%, respectively.

Table 2. Different methods of comparison

Method		RMSE	MAE	R ²	Method		RMSE	MAE	R ²
NSW electricity demand	SVM	88.28	65.52	.97	VIC electricity demand	SVM	68.82	51.39	.98
	RT	100.27	75.09	.97		RT	78.40	60.02	.98
	LSTM	58.21	40.51	.99		LSTM	41.44	30.69	.99
NSW electricity price	SVM	32	9.78	.49	VIC electricity price	SVM	48.23	12.76	.55
	RT	41.96	12.47	.12		RT	39.56	13	.53
	LSTM	29.67	7.36	.78		LSTM	33.46	10.76	.88

7 Conclusion

COVID-19 pandemic has significantly influenced people's lifestyle in many ways. First, in two scenarios, the implications of the pandemic are being analysed from comprehensive perspectives on electricity demand and price data. The results show that different restriction measures in both scenarios and their impact on people's activities have considerably changed the electricity demand and price profile distinctively. For example, NSW with the highest population and less restrictive measures has more demand and price reduction than VIC. Electricity demand has increased in July 2020 for both scenarios compared to the same period in 2019. Significant price reductions were observed on weekends by 48% for NSW and 47% for VIC due to lower demand, increased renewable output, and lower oil prices. During the pandemic, overall demand and price patterns have not changed significantly compared to the same period last year even the total electricity production has dropped beside the demand. Residential demand has increased due to shut-down, working from home setup, and winter weather; and commercial demand has reduced due to limited business activities. Since residential demand is related to weather than other industries and based on the Practice Theory [23], people's activities at resident would be expressed as routine recurrent trends in terms of electricity usage, regardless of irregularity. Therefore, secondly, this paper proposes a LSTM based sequence-to-sequence network model to forecast electricity demand and price considering the current uncertain pandemic situation. This model can manage variable input and output length, and effectively learns temporal correlation in the input sequence to model temporal structure simultaneously. A few traditional forecast models are comprehensively tested and compared to the proposed model on real market data. Simulation results prove the effectiveness of the proposed method over others with smaller errors and good accuracy despite its conceptual simplicity. Future work will be focused on the severity and long term effect of the COVID-19 pandemic on the electricity market since we are still experiencing the pandemic.

References

1. Adhikari, R., Agrawal, R.K.: An introductory study on time series modeling and forecasting. arXiv preprint [arXiv:1302.6613](https://arxiv.org/abs/1302.6613) (2013)
2. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. *Adv. Neural Inf. Process. Syst.* **27**, 3104–3112 (2014)
3. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436–444 (2015)
4. Mikolov, T., Joulin, A., Chopra, S., Mathieu, M., Ranzato, M.A.: Learning longer memory in recurrent neural networks. arXiv preprint [arXiv:1412.7753](https://arxiv.org/abs/1412.7753) (2014)
5. Graves, A., Mohamed, A.-R., Hinton, G.: Speech recognition with deep RNN networks. In: *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649 (2013)
6. Nelson, D.M., Pereira, A.C., de Oliveira, R.A.: Stock market's price movement prediction with LSTM neural networks. In: *IEEE International Joint Conference on Neural Networks (IJCNN)*, pp. 1419–1426 (2017)
7. Le, X.-H., Ho, H.V., Lee, G., Jung, S.: Application of long short-term memory (LSTM) neural network for flood forecasting. *Water* **1**(7), 1387 (2019)

8. Ugurlu, U., Oksuz, I., Tas, O.: Electricity price forecasting using recurrent neural networks. *Energies* **11**(5), 1255 (2018)
9. Lipton, Z.C., Berkowitz, J., Elkan, C.: A critical review of recurrent neural networks for sequence learning. arXiv preprint [arXiv:1506.00019](https://arxiv.org/abs/1506.00019) (2015)
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
11. Bouktif, S., Fiaz, A., Ouni, A., Serhani, M.A.: Optimal deep learning LSTM model for electric load forecasting using feature selection and genetic algorithm: comparison with machine learning approaches. *Energies* **11**(7), 1636 (2018)
12. Li, C., Ding, Z., Zhao, D., Yi, J., Zhang, G.: Building energy consumption prediction: an extreme deep learning approach. *Energies* **10**(10), 1525 (2017)
13. Tong, C., Li, J., Lang, C., Kong, F., Niu, J., Rodrigues, J.J.: An efficient deep model for day-ahead electricity load forecasting with stacked denoising auto-encoders. *J. Parallel Distrib. Comput.* **117**, 267–273 (2018)
14. Marino, D.L., Amarasinghe, K., Manic, M.: Building energy load forecasting using deep neural networks. In: *IECON 2016–42nd Annual Conference of the IEEE Industrial Electronics Society*, pp. 7046–7051 (2016)
15. Sehovac, L., Grolinger, K.: Deep learning for load forecasting: sequence to sequence recurrent neural networks with attention. *IEEE Access* **8**, 36411–36426 (2020)
16. Gong, G., An, X., Mahato, N.K., Sun, S., Chen, S., Wen, Y.: Research on short-term load prediction based on Seq2seq model. *Energies* **12**(16), 3199 (2019)
17. Kuo, P.-H., Huang, C.-J.: An electricity price forecasting model by hybrid structured deep neural networks. *Sustainability* **10**(4), 1280 (2018)
18. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**(1), 1929–1958 (2014)
19. Aggregated Demand and Price Data. <https://aemo.com.au/>, Accessed 28 Sept 2020
20. Hu, M.Y., Zhang, G., Jiang, C.X., Patuwo, B.E.: A cross-validation analysis of neural network out-of-sample performance in exchange rate forecasting. *Decis. Sci.* **30**(1), 197–216 (1999)
21. Garreta, R., Moncecchi, G.: *Learning Scikit-Learn: Machine Learning in Python*. Packt Publishing Ltd., Birmingham (2013)
22. National Electricity Market. <https://opennem.org.au/energy/nem/>, Accessed 26 Sept 2020
23. Stephen, B., Tang, X., Harvey, P.R., Galloway, S., Jennett, K.I.: Incorporating practice theory in sub-profile models for short term aggregated residential load forecasting. *IEEE Trans. Smart Grid.* **8**(4), 1591–1598 (2015)



Cross-database Micro Expression Recognition Based on Apex Frame Optical Flow and Multi-head Self-attention

Jiebin Wen^{1,2}, Wenzhong Yang^{1,2}(✉), LieJun Wang^{1,2}, Wenyu Wei^{1,2}, Sixiang Tan^{1,2}, and Yongzhi Wu^{1,2}

¹ College of Information Science and Engineering,
Xinjiang University, Urumqi, Xinjiang, China
ywz_xy@163.com

² Key Laboratory of Multilingual Information Technology in Xinjiang Uygur
Autonomous Region, Xinjiang University, Urumqi, Xinjiang, China

Abstract. With the rise of deep learning in the field of compute vision, more and more researchers began to use CNNs for automatic recognition of micro-expressions, and achieved better performance than traditional methods. However, there is a overfitting problem with these methods. CNN focuses on local information and loses the global semantic information in the image. In this paper, we propose a novel micro-expression recognition method based on Apex frame and multi-head self-attention enhanced convolutional network. It consists of three parts: (1) The pre-processing part performs face detection, face alignment, cutting into uniform size and positioning Apex frames on micro-expressions; (2) Optical flow calculation of Apex frames, calculates TVL1 optical flow from Onset frames to Apex frames, and obtains horizontal and vertical optical flow component images; (3) A shallow network (AACNet) is designed to extract and classify the optical flow image components obtained in (2). The results has greatly improved over the benchmark method (LBP-TOP). The state-of-the-art results were achieved on the MEGC2019 database (UF1: 0.7572, UAR: 0.7564).

Keywords: Micro-expression recognition · Apex frame · Optical flow · Multi-head self-attention

1 Introduction

Micro-expression [1] is a spontaneous facial expression and usually occur when people try to hide their true emotions, and they cannot be faked or suppressed [2]. Micro-expressions can reflect people's true emotions, and have potential applications in criminal investigation trials, marriage relationship prediction, national security and other fields.

With the development of computer vision, more and more researchers began to use machine learning methods to automatically recognize micro-expressions. The micro-expression automatic recognition system usually consists of three stages: preprocessing, feature extraction and classification [3]. How to design low-dimensional features without losing key information is the key of feature extraction. According to the different characteristics, the mainstream methods include LBP-based methods (4) and optical flow-based methods [5–7]. However, due to the short duration and low intensity of micro-expressions, it is difficult to find suitable feature descriptors, so the recognition rate of traditional manual feature methods is still not high.

Recently, many researchers began use deep features to represent micro-expressions. The deep neural network methods require a large amount of data for training, but the number of samples in the micro-expression dataset is limited. The problem limits the potential of the deep learning method to a certain extent. In this paper we propose a micro-expression recognition method that combines traditional optical flow features and self-attention-enhanced convolutional network. In a limited data sample, self-attention is used to capture the internal correlation information of micro-expression movements.

2 Related Work

2.1 Micro-expression Recognition Based on Apex Frame Optical Flow

In most studies, micro-expressions are treated as a video sequence. The conventional feature extraction methods consider the entire video sequence or the partial sequence processed by the Temporal Interpolation Model [8] (TIM). Researchers represented by Liong et al. [7] believe that the micro-expression sequence at high frame rate is not necessary for every frame, and on the contrary it may bring some computational redundancies. They proposed a feature extraction method (Bi-WOOF) that uses only two images (starting frame and peak frame) to represent micro expressions. And proposed a Divide & Conquer [7] algorithm to achieve Apex frame positioning, and then calculated the optical flow image from Onset frame (reference frame) to the Apex frame.

Apex frames represent the peak of facial micro-expression movement and contain very discriminative features. Using the Apex frame optical flow method can reduce the amount of calculation, and a large number of studies have indeed verified the excellent performance of this method.

2.2 Micro-expression Recognition Based on Deep Learning

Deep learning has made great progress in the field of computer vision. Many researchers use end-to-end CNN and LSTM networks to extract the spatial and temporal features of micro-expression sequences. In recent years, there have been a large number of studies on the use of convolutional neural networks for micro-expression recognition. According to the different network architectures, they

can be roughly divided into 3DCNN structure, recursive convolutional network structure, and dual-stream network structure.

The micro-expression recognition method based on deep learning has achieved better performance than manual methods. However, deep learning methods usually require a lot of data for training. Due to the lack of micro-expression samples, over-fitting problems often occur, making it difficult for the recognition rate of micro-expression to continue to increase. In order to solve the problem of model overfitting caused by insufficient micro-expression datasets, many studies have adopted the method of transfer learning [9–12] for micro-expression recognition.

2.3 Cross-Database Micro Expression Recognition

Due to insufficient samples in the datasets, some scholars try to combine the datasets to increase the number of micro-expression samples. The public cross-database micro-expression dataset was first introduced in the MEGC2018 [13]. CASMEII [14] and SAMM [15] were recombined into five expression categories. In the second MEGC Challenge [16], the SIMC [4] dataset was added to the fusion dataset. In order to make all three datasets usable together, a common set of simplified emotion categories (positive, negative and surprised) were used appropriately mapping the original sentiment category. Peng et al. [9] used the transfer learning method, using ImageNet and the trained ResNet10 model to first train on five macro expression datasets, and then use the micro expression dataset for fine-tuning after reaching an accuracy of 99.35

Cross-database datasets can fit real scenes more realistically. Firstly, it increases the number of objects, and objects come from more regions, races, and shooting environments. More micro-expression samples are more conducive to data-driven deep learning methods; secondly, the reduced conventional emotion categories are used to better adapt to the contrasting types of emotions caused by different stimuli and environmental settings (Fig. 1).

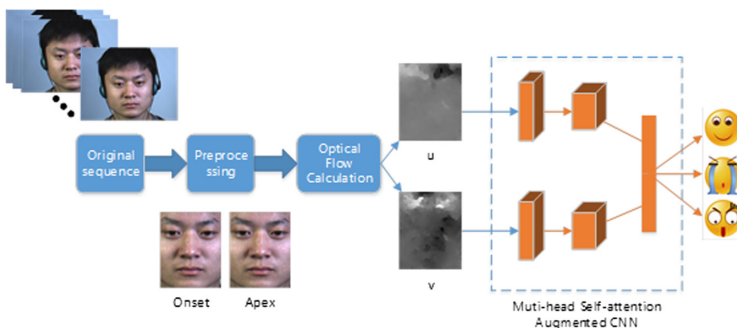


Fig. 1. Framework of the proposed method

3 The Proposed Method

3.1 Preprocessing

The preprocessing part includes facial feature point positioning and face alignment and Apex frame positioning. Among them, SMIC and CASMEII have provided cropped micro expression sequences. For the SAMM dataset, we use the Face++ [17] API to extract facial feature points, and then use Dlib [18] to perform face alignment based on the extracted feature points. The Apex frame represents the peak of the motion intensity of the micro expression, and is the frame with the largest change. CASME II and SAMM datasets have provided expert Apex frame annotations, and we directly use the annotations as Apex frames. For the SMIC dataset that does not provide Apex frame annotations, we directly use the D&C-RoIs [7] algorithm proposed by Li et al. to locate Apex frames.

3.2 Optical Flow Calculation

Optical flow is the instantaneous velocity of the pixel movement of a space moving object on the observation imaging plane. The optical flow method uses the correlation between adjacent frames to find the correspondence between the previous frame and the current frame, thereby calculating the movement information of objects between adjacent frames. Generally, the instantaneous change rate of gray scale at a specific coordinate point on a two-dimensional image plane is defined as an optical flow vector.

The definition of optical flow must satisfy two basic assumptions: (1) The brightness is constant. That is, when the same target moves between different frames, its brightness will not change. (2) The time is continuously short or the exercise is “small exercise”. That is, changes in time will not cause drastic changes in the target position, and the displacement between adjacent frames should be relatively small. The magnitude of micro-expression change is low and local, and the duration of the micro-expression is very short, so, the micro-expression meets the two basic assumptions of optical flow.

Consider the light intensity of a pixel $I(x, y, t)$ in the starting frame (where t represents its time dimension). It moved the distance (dx, dy) to the peak frame and took dt time. According to the first assumption mentioned above, we believe that the light intensity of the pixel is constant, namely:

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \quad (1)$$

Carrying out the first-order Taylor expansion on the right end of (1):

$$I(x, y, t) = I(x, y, t) + \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + \frac{\partial I}{\partial t} dt + \varepsilon \quad (2)$$

Where ε represents the second-order infinitesimal term, which can be ignored. Then divide (2) generation (1) after the same by dt , we can get:

$$\frac{\partial I}{\partial x} \frac{dx}{dt} + \frac{\partial I}{\partial y} \frac{dy}{dt} + \frac{\partial I}{\partial t} \frac{dt}{dt} = 0 \quad (3)$$

Suppose u and v are the velocity vectors along the X axis and the Y axis respectively, and we get:

$$u = \frac{dx}{dt}; v = \frac{dy}{dt} \quad (4)$$

Let $I_x = \frac{\partial I}{\partial x}$, $I_y = \frac{\partial I}{\partial y}$, $I_t = \frac{\partial I}{\partial t}$ denote the partial derivatives of the gray levels of the pixels in the image along the X, Y, and T directions. In summary, formula (3) can be written as:

$$I_x u + I_y v + I_t = 0 \quad (5)$$

Among them, I_x , I_y , I_t can all be obtained from the image data, and (u, v) is the optical flow vector of the Apex frame to be obtained. Liong et al. [19] verified the impact of five optical flow algorithms on micro-expression recognition, and the results proved that the optical flow algorithm of TVL1 [20] can achieve the best results in micro-expression recognition. Therefore, this paper adopts the TVL1 optical flow algorithm, uses the OpenCV library to calculate the optical flow, and returns the horizontal and vertical components of the result.

3.3 Multi-head Self-attention Enhanced CNN

The self-attention mechanism is an improvement of the attention mechanism, which reduces the dependence on external information and is better at capturing the internal correlation of data or features. Ashish et al. [21] proposed a triplet (key, query, value) to capture long-distance dependence. Attention can be described as mapping a query and a set of key-value pairs to the output. The query, key, value and output are all vectors. The output is calculated as a weighted sum of values, where the weight assigned to each value is calculated by querying the compatibility function with the corresponding key. Calculate the dot product of the query using all the keys, divide each key by $\sqrt{d_k}$, and then apply the Softmax function to get the weight of the value.

$$Attention(Q, K, V) = Softmax\left(\frac{(QK)^T}{\sqrt{d_k}}\right)V \quad (6)$$

Self-attention enhanced convolution is shown in Fig. 3, which combines ordinary convolution and multi-head attention to ensure that the extracted features contain local and global information. H , W , F_{in} represent the height, width, and number of filters of the feature map, and N_h , d_k and d_v represent the number of heads, key and query values, respectively. Assume that N_h is divisible by d_k and d_v , d_k^h and d_v^h is the depth of each head key and query (Fig. 2).

Given an input tensor (H, W, F_{in}) , convert it into a matrix $X \in \mathbb{R}^{(HW \times F_{in})}$, and then execute the multi-head attention [21] in formula (6), the output of a head can be written as

$$O_h = Softmax\left(\frac{(XW_q)(XW_k)^T}{\sqrt{d_k^h}}\right)(XW_v) \quad (7)$$

Where $W_q, W_k \in \mathbb{R}(F_i n \times d_k^h)$, $W_v \in \mathbb{R}(F_m \times d_v^h)$ are the linear transformations from input X to queries Q, keys K, and values V, respectively.

The output of all heads MHA is

$$MHA(x) = Concat[O_1, \dots, O_{N_h}] \tag{8}$$

Where $W^O \in \mathbb{R}^{d_v \times d_v}$ is a linear transformation, and then MHA(X) is reshaped into (H, W, F_{in}) . The last step is to merge the multi-head attention feature map and the convolution feature map, and the output of the self-attention enhanced convolution AACConv(X) is

$$AACConv(X) = Concat[Conv(X), MHA(X)] \tag{9}$$

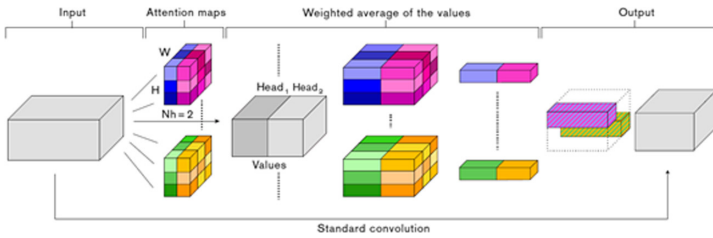


Fig. 2. Self-attention augmented convolution

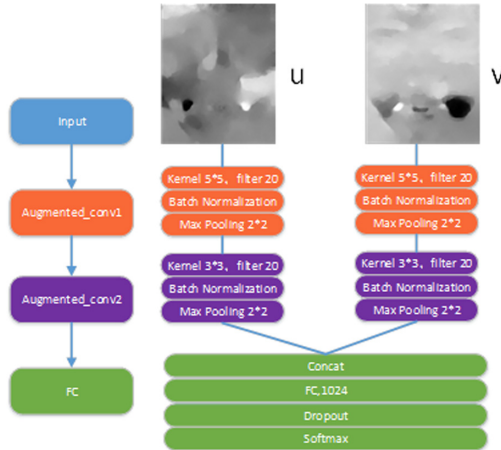


Fig. 3. Multi-head self-attention augmented CNN

The self-attention enhanced convolutional network used in this paper is shown in Fig. 3. In order to avoid the over-fitting problem caused by insufficient

data samples, a shallow network design is adopted, including 2 self-attention enhanced convolutional layers. The network input is the Apex frame optical flow components u and v , which are enhanced convolution Augmented.conv through two layers of self-attention, each layer includes enhanced convolution, batch normalization, and maximum pooling. The first layer of convolution kernel size is 55, the second layer of convolution kernel size is 33, and the number of output channels is 20. The output of the second layer of the two components is spliced and then connected to a fully connected layer. After the fully connected layer, a dropout layer is added to reduce the number of parameters. Finally, softmax is used to complete the micro-expression classification task.

4 Experiment

4.1 Database

In order to compare with the classic methods and the current SOTA methods, the experiment uses the MEGC2019 fusion dataset for verification. The fusion dataset is composed of SMIC [4], CASME II [14] and SAMM [15]. SMIC recorded a total of 20 subjects and found 164 micro-expression fragments from 16 subjects. The three emotion categories are “positive” and “Negative” and “surprised”. The CASMEII dataset includes 247 micro-expression samples from 35 subjects and provides 5 emotion annotations: “happy”, “disgust”, “surprised”, “repressed” and “other”. SAMM collected the micro expressions of 32 subjects from different groups of people, including 159 micro expression clips, and 7 emotion categories are marked: “happy”, “surprised”, “sad”, “anger”, “fear”, “disgust”, “contempt”. The fusion dataset is unified into three categories (“positive”, “negative” and “surprise”), among which the “happy” category is relabeled as “positive”; “disgust”, “depressed”, “anger”, “contempt” and “fear” were relabeled as “negative”; the “surprise” category remained unchanged. The fused dataset contains 442 micro expression sequences from 68 subjects. The specific quantities from each dataset are detailed in Table 1.

Table 1. Summary of combined micro-expression database

	Subjects	Positive	Negative	Surprise	Total
SMIC	16	51	70	43	164
CASME II	24	32	88	25	145
SAMM	28	26	92	15	132
Combined	68	109	250	83	442

4.2 Performance Metric

In this paper we use the LOSOCV (Leave-One-Subject-Out Cross-Validation) experimental protocol. The fusion dataset contains a total of 68 subjects, so the experiment is divided into 68 folds. For each fold, the micro-expression sequence of one object is selected as the test set, and all the micro-expression sequences of the remaining 67 objects are used as the training set. The evaluation standard uses Unweighted F1-score (UF1) and Unweighted Average Recall (UAR). UF1 is a good choice in multi-classification problems because it can emphasize rare classes equally. In order to calculate UF1, first calculate the number of true positive (TP), false positive (FP) and false negative (FN) of each category C of the k -th compromise in LOSO and calculate the F1 value of this category. UF1 is the average value of F1 for each category.

$$F1_c = \frac{2TP_c}{2TP_c + FP_c + FN_c} \quad (10)$$

$$UF1 = \frac{1}{C} \sum_C F1_c \quad (11)$$

UAR is also called “balanced accuracy rate”, which is a more reasonable evaluation standard that replaces the standard accuracy rate (or weighted average recall) because its predictions are more biased towards larger categories of results.

$$UAR = \frac{1}{C} \sum_C Acc_c \quad (12)$$

$$ACC_c = \frac{TP_c}{N_c} \quad (13)$$

Where C is the number of categories, and N_c is the total number of samples in category c .

4.3 Apex Frame Images

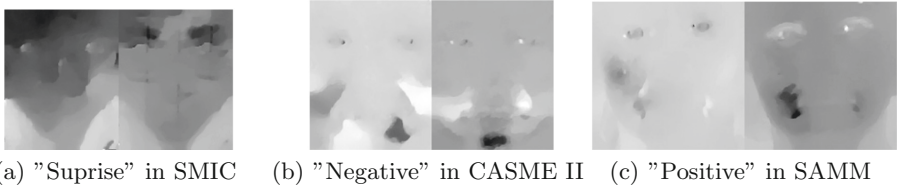


Fig. 4. Samples of Apex frame optical flow images

We use the OpenCV library to calculate the TVL1 optical flow between Onset and Apex. The effect is shown in Fig. 4, which is the horizontal and vertical

components of the three types of optical flow. From the generated optical flow diagram, the contour and the change range of the human face can be roughly seen. The areas of local motion changes for different micro-expression categories are different. It can be seen from Fig. 4 that the optical flow of TVL1 extracted in this paper can reflect the movement characteristics of the micro-expression.

4.4 Experiments Settings

The self-attention-enhanced convolutional network is implemented using Keras. The input of the horizontal and vertical optical flow images are normalized to 2828 pixels. After two attention-enhanced convolutional layers, the output dimension is 7720, stitching the latter dimension is 1960. After a 1024-dimensional fully connected layer and a dropout layer with a dropout probability of 0.5, softmax is used to divide into 3 categories. The specific attention enhanced convolution module structure is shown in Fig. 4, 20 filters, the number of heads of multi-head attention num_heads is 4, depth_v and depth_k are both 4, the number of channels of ordinary convolution is 16, and the two are connected. The number of output channels is still 20.

The experiment uses the LOSOCV protocol, the dataset is divided into 68 folds, for each fold of training data, epoch is set to 40, batch_size is set to 8, and the learning rate is 0.0001. Use categorical_crossentropy as the loss function, and the optimizer is Adam.

4.5 Analysis and Discussion

Table 2 shows the experimental results of LOSOCV using self-attention enhanced convolutional networks, and the LBP-TOP method is the benchmark method. SA-AT [12], ATNet [23], CapsuleNet [24], Dual-Inception [25] and STSTNet [26] are five representative deep learning methods.

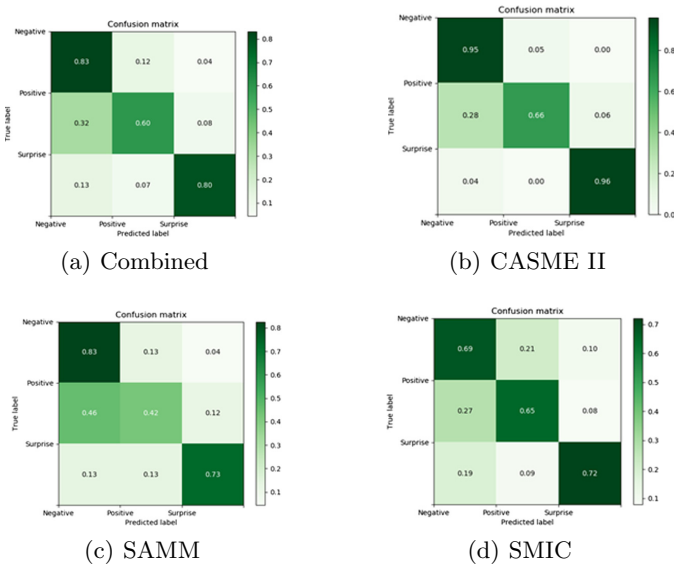
It can be clearly seen that the proposed cross-database micro-expression recognition method based on self-attention enhanced convolution has greatly improved compared with the benchmark method, with UF1 reaching 0.7572 and UAR reaching 0.7564. It is worth noting that UF1 and UAR work best on the CASME II dataset. The poor performance of the SMIC dataset may be due to low resolution and frame rate. The problem in the SAMM dataset may be the imbalance of the number of samples between categories. Therefore, there are still greater challenges in the identification of SMIC and SAMM datasets. In addition, the confusion matrix is shown in Fig. 5.

In order to further analyze the performance difference of the AACNet method in different datasets, a confusion matrix of the fusion dataset and the classification results of three separate datasets is drawn. As shown in Fig. 5, (a–d) represent the confusion matrix of the fusion dataset, CASME II, SAMM and SMIC respectively. The ordinate represents the true label of the micro-expression, the abscissa represents the predicted classification label, and the value represents the predicted probability of each category. For example, 0.83 in the fusion dataset represents the probability that the actual category is “negative” and is correctly

Table 2. UF1 and UAR comparison

Method	FULL		SMIC		CASME II		SAMM	
	UF1	UAR	UF1	UAR	UF1	UAR	UF1	UAR
LBP-TOP(Baseline)	0.5882	0.5785	0.2000	0.5280	0.7026	0.7429	0.3954	0.4102
Bi-WOOF	0.6296	0.6227	0.5727	0.5829	0.7805	0.8026	0.5211	0.5139
SA-AT [12]	0.5936	0.5958	0.5512	0.5463	0.7607	0.7552	0.4476	0.4868
ATNet [23]	0.6310	0.6130	0.5530	0.5430	0.7980	0.7550	0.4960	0.4820
CapsuleNet [24]	0.6520	0.6506	0.5820	0.5877	0.7068	0.7018	0.6209	0.5989
Dual-Inception [25]	0.7322	0.7278	0.6604	0.6726	0.8621	0.8560	0.5868	0.5663
STSTNet [26]	0.7353	0.7605	0.6801	0.7013	0.8382	0.8686	0.6588	0.6810
AACNet(Ours)	0.7572	0.7564	0.7067	0.7077	0.8656	0.8569	0.6726	0.6847

predicted as “negative”. It can be seen that many “positive” or “surprised” categories are incorrectly predicted as “negative”. The reason may be that the categories are unbalanced. The total number of “negative” samples exceeds half (250/442), so the classification results are biased towards “negative” category. The poor performance of the SMIC dataset may be due to the lower resolution and frame rate. The problem in the SAMM dataset may be the imbalance in the number of samples between categories. The number of “surprised” and “positive” samples accounted for only 10% and 20% of the total, respectively. Therefore, there are still great challenges in the identification of SMIC and SAMM datasets.

**Fig. 5.** Confusion matrix of AACNet

5 Conclusion

This paper proposes a micro-expression recognition method combining Apex optical flow and self-attention enhanced convolutional neural network. For the micro-expression sequence, only the Onset frame and the Apex frame are used, and the optical flow components are extracted and classified through the end-to-end network AACNet. Experimental results show that the performance of the proposed cross-database micro-expression recognition based on self-attention-enhanced convolution is significantly better than benchmark methods and some excellent deep learning methods. The dataset used in this experiment is a fusion dataset reorganized on the basis of 3 typical datasets, which not only increases the number of subjects and samples, but also increases the diversity of subjects' race, age and other characteristics and sample diversity. In addition, the LOSO verification method is adopted to ensure that the objects are independently evaluated and conform to the real application scenarios, so the algorithm has strong robustness and practicality. The recognition performance of this method on the SAMM and SMIC databases is obviously inferior to the CASME II dataset. In future work, data augmentation methods such as GAN (Generative Adversarial Network) can be considered to increase the number of datasets and sample balance between categories.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China (No. U1603115), National key R&D plan project (2017YF C0820702-3) and National Engineering Laboratory for Public Safety Risk Perception and Control by Big Data (XJ201810101).

References

1. Ekman, P., Friesen, W.V.: Nonverbal leakage and clues to deception. *Psychiatry* **32**(1), 88–106 (1969)
2. Ekman, P.: Darwin, deception, and facial expression. *Ann. N. Y. Acad. Sci.* **1000**(1), 205–221 (2003)
3. Gan, Y.S., Liong, S.T., Yau, W.C., et al.: Off-apexnet on micro-expression recognition system. *Sig. Process. Image Commun.* **74**, 129–139 (2019)
4. Pfister, T., Li, X., Zhao, G., et al.: Recognising spontaneous facial micro-expressions. In: 2011 International Conference on Computer Vision, pp. 1449–1456. IEEE (2011)
5. Liu, Y.J., Zhang, J.K., Yan, W.J., et al.: A main directional mean optical flow feature for spontaneous micro-expression recognition. *IEEE Trans. Affect. Comput.* **7**(4), 299–310 (2015)
6. Xu, F., Zhang, J., Wang, J.Z.: Microexpression identification and categorization using a facial dynamics map. *IEEE Trans. Affect. Comput.* **8**(2), 254–267 (2017)
7. Liong, S.T., See, J., Wong, K.S., et al.: Less is more: micro-expression recognition from video using apex frame. *Sig. Process. Image Commun.* **62**, 82–92 (2018)
8. Zhou, Z., Zhao, G., Pietikäinen, M.: Towards a practical lipreading system. In: CVPR 2011, pp. 137–144. IEEE (2011)

9. Peng, M., Wu, Z., Zhang, Z., et al.: From macro to micro expression recognition: deep learning on small datasets using transfer learning. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), pp. 657–661. IEEE (2018)
10. Wang, S.J., Li, B.J., Liu, Y.J., et al.: Micro-expression recognition with small sample size by transferring long-term convolutional neural network. *Neurocomputing* **312**, 251–262 (2018)
11. Zhi, R., Xu, H., Wan, M., et al.: Combining 3D convolutional neural networks with transfer learning by supervised pre-training for facial micro-expression recognition. *IEICE Trans. Inf. Syst.* **102**(5), 1054–1064 (2019)
12. Zhou, L., Mao, Q., Xue, L.: Cross-database micro-expression recognition: a style aggregated and attention transfer approach. In: 2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 102–107. IEEE (2019)
13. Yap, M.H., See, J., Hong, X., et al.: Facial micro-expressions grand challenge 2018 summary. In: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), pp. 675–678. IEEE (2018)
14. Yan, W.J., Li, X., Wang, S.J., et al.: CASME II: an improved spontaneous micro-expression database and the baseline evaluation. *PLoS ONE* **9**(1), e86041 (2014)
15. Davison, A.K., Lansley, C., Costen, N., et al.: SAMM: a spontaneous micro-facial movement dataset. *IEEE Trans. Affect. Comput.* **9**(1), 116–129 (2016)
16. See, J., Yap, M.H., Li, J., et al.: MEGC 2019-the second facial micro-expressions grand challenge. In: 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), pp. 1–5. IEEE (2019)
17. Megvii, I.: Face++ research toolkit (2013)
18. King, D.E.: Dlib-ml: a machine learning toolkit. *J. Mach. Learn. Res.* **10**(Jul), 1755–1758 (2009)
19. Liong, S.T., Gan, Y.S., Zheng, D., et al.: Evaluation of the spatio-temporal features and GAN for micro-expression recognition system. *J. Signal Process. Syst.* 1–21 (2020)
20. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime TV- L^1 optical flow. In: Hamprecht, F.A., Schnörr, C., Jähne, B. (eds.) DAGM 2007. LNCS, vol. 4713, pp. 214–223. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74936-3_22
21. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: *Advances in Neural Information Processing Systems*, pp. 5998–6008 (2017)
22. Bello, I., Zoph, B., Vaswani, A., et al.: Attention augmented convolutional networks. In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3286–3295 (2019)
23. Peng, M., Wang, C., Bi, T., et al.: A novel apex-time network for cross-dataset micro-expression recognition. In: 2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII), pp. 1–6. IEEE (2019)
24. Van Quang, N., Chun, J., Tokuyama, T.: CapsuleNet for micro-expression recognition. In: 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), pp. 1–7. IEEE (2019)
25. Zhou, L., Mao, Q., Xue, L.: Dual-inception network for cross-database micro-expression recognition. In: 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), pp. 1–5. IEEE (2019)
26. Liong, S.T., Gan, Y.S., See, J., et al.: Shallow triple stream three-dimensional CNN (STSTNet) for micro-expression recognition. In: 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), pp. 1–5. IEEE (2019)



GPS Intelligent Solution of Aerial Image Target in State Grid EIA Survey

Yu Wu¹, Fei Wang², Caihua Sun¹, Songyang Zhang¹, Jie Huang³,
Zhentao Liu⁴, and Wei Sun^{1,3(✉)}

¹ State Grid He-Nan Electric Power Company, Zhengzhou, China
wsun@xidian.edu.cn

² Henan Jiuyu Enpai Power Technology Co., Ltd., Zhengzhou, China
³ Xidian University, Xi'an, China

⁴ Xi'an University of Posts and Telecommunications, Xi'an, China

Abstract. This article is mainly aimed at defining and verifying whether the project involves environmentally sensitive areas in the process of environmental intelligent survey, calculating the position coordinate information of the target GPS position in the aerial image map under the condition of Gaussian projection, and further calculating the four corners of the image and the position of other target points under the WGS84 coordinate system. First of all, based on the POS data information of aerial equipment, this paper calculates the corresponding Gaussian projection and POS data information, then the pseudo-position information of image center coordinates is further calculated; Second, Based on the coordinates information of gauss projection, this paper calculates the heading open angle and side open angle of the camera, then calculate the width and height of the image for pose correction, and finally get the coordinate position of the corrected image center point. Finally, the GPS positions of other target points can be calculated by pixel coordinates of images, and the calculated results are pasted on Google earth for precision comparison. Experimental results show that the proposed method can accurately calculate the GPS positions of targets in aerial images.

Keywords: Environmental exploration · GPS · Aerial images · Gauss projection · Coordinate correction

1 Introduction

With the continuous improvement of the global economy, people's awareness of environmental protection is also gradually improving. Environmental impact assessment [1–3] and environmental monitoring have become important tasks in the current era, which can supervise and manage huma's damage to nature and reduce the harm to nature. People should fully consider the environmental impact in the project implementation process, improve the environment and strengthen the environment through the modern advanced scientific and technological achievements protection [4–6]. Therefore, in the process of project implementation, project planning route should be considered the impact on the ecological protection.

At present, in the EIA of the project, it is still necessary to include the sensitive elements and sensitive targets in the project into the EIA index objects and make corresponding data sets to provide the basis for the EIA work. In environmental exploration projects, EIA work mainly includes two categories: 1) target detection of EIA; 2) illegal building records [7, 8]. At present, the detection accuracy, target position accuracy and position accuracy of illegal buildings in EIA work are not enough. This paper mainly aims at solving the position coordinate information of GPS position [9] in aerial image [10–12] under the condition of Gaussian projection [13] [15], and further calculate the position coordinates of four corners of the image under WGS84.

2 Aerial Photography Platform

The fixed-wing UAV is the main platform for data collection, as shown in Fig. 1.



Fig. 1. Fixed-wing UAV

After obtaining POS data from the POS system of aerial photography equipment, through projection calculation and attitude correction, the position coordinates of corresponding image center points in WGS84 coordinate system are output, and the position coordinates of other targets can be obtained. The main process is shown in Fig. 2.

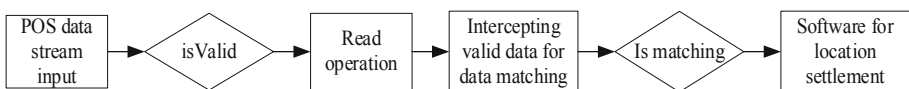


Fig. 2. Main flow chart

3 Algorithm

3.1 Calculation of Image Center Point

In this paper, by obtaining POS data information of aerial photography equipment, the coordinate position information of image pseudo-center point is calculated. After attitude correction [18, 19], the coordinate positions of the image center point were obtained. Then the coordinates of corner points and other sensitive points can be calculated from the pixel coordinates of the image. As shown in Fig. 3, where heading Angle κ and elevation Angle α . The projection coordinate of the photography point is (X_0, Y_0) , and the center coordinate of the image after pose correction is (X, Y) .

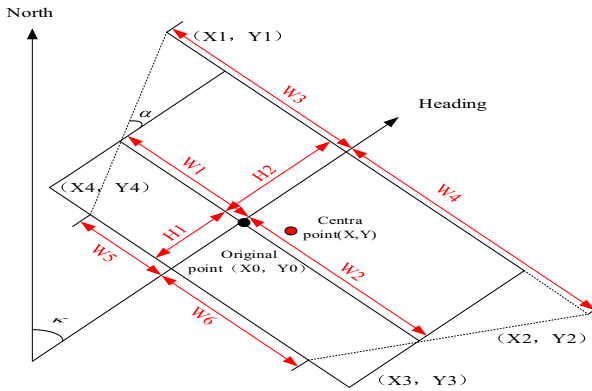


Fig. 3. Schematic diagram of aerial image

In Fig. 3, the coordinates of the center point of the image and the four corner points are calculated. The required POS data information is as follows: longitude and latitude, altitude A , pitch Angle, yaw Angle, and heading Angle. In addition, it is necessary to know the focal length of the aerial photography equipment and the width and height of the image.

Firstly, the plane coordinates (X_0, Y_0) from longitude and latitude information (L, B) to WGS84 geodetic coordinates are calculated by Gauss projection [16] forward formula:

$$X_0 = X + \frac{f^2}{2} N \sin B \cos B + \frac{f^2}{24} N \sin B \cos^3 B (5 - t^2 + 9\eta^2 + 4\eta^2) + \frac{f^6}{720} N \sin B \cos^5 B (61 - 58t^2 + t^4) \quad (1)$$

$$Y_0 = lN \cos B + \frac{f^3}{6} N \cos^3 B (1 - t^2 + \eta^2) + \frac{f^5}{120} N \cos^5 B (5 - 18t^2 + t^4 + 14\eta^2 - 58\eta^2 t^2) \quad (2)$$

In the formula, B is the latitude of the earth when the UAV is shooting instantaneously, $l = L - L_0$ is the longitude difference between the image point and the central

meridian, where L is the earth longitude during shooting, and L_0 is the longitude of the central meridian; the calculation formula of the meridian arc length from the equator to the dimension B can be written as follows:

$$X = a_0 B - \sin B \cos B \left[(a_2 - a_4 + a_6) + \left(2a_4 - \frac{16}{3} \right) \sin^2 B + \frac{16}{3} a_6 \sin^4 B \right] \quad (3)$$

$N = a(1 - e^2 \sin^2 B)^{1/2}$ is the radius of curvature of the meridian circle, $t = \tan B$, $\eta = e' \cos B$, $e' = \frac{\sqrt{a^2 - b^2}}{b}$, a is the long half axis of the ellipsoid, $b = a(1 - f)$ is the short half axis of the ellipsoid, f is the ellipsoid flatness.

$$\begin{aligned} a_0 &= m_0 + m_2/2 + 3/8 m_4 + 5m_6/16 + 35m_8/128 & a_4 &= m_4/8 + 3m_6/16 + 7m_8/32 \\ a_2 &= m_2/2 + m_4/2 + 15m_6/32 + 7m_8/16 & a_6 &= m_6/32 + m_8/16 \\ a_8 &= m_8/128 \end{aligned}$$

In the formula,

$$m_0 = a(1 - e^2); \quad m_2 = \frac{3}{2} e^2 m_0; \quad m_4 = 5e^2 m_2; \quad m_6 = \frac{7}{6} e^2 m_4; \quad m_8 = \frac{9}{8} e^2 m_6 \quad (4)$$

Second, the side angle, pitch angle and heading angle of the center point need to be corrected. From the pose, the correction formula of the image center in WGS84 can be obtained

$$\begin{aligned} X_a &= A \tan \phi \sin \kappa && \text{pitch correction} \\ X_b &= A \tan \omega \cos \kappa && \text{correction of lateral deviation} \\ Y_a &= A \tan \phi \cos \kappa && \text{pitch correction} \\ Y_b &= A \tan \omega \sin \kappa && \text{correction of lateral deviation} \end{aligned}$$

Then the coordinates (x, y) of the image center point can be written as:

$$\begin{aligned} X &= X_0 + X_a + X_b \\ Y &= Y_0 + Y_a - Y_b \end{aligned} \quad (5)$$

3.2 Calculation of Corner Coordinates

After calculating the coordinates of the center point of the image, the position coordinates of the four corner points can be calculated according to the ratio of the pixel and the map frame, and then the coordinates of the four corner points of the image frame are calculated. The calculation of side opening angle δ and heading opening angle θ is determined by formula (5) and Eq. (6) respectively

$$\theta = \arctan\left(\frac{W_0}{2F}\right) \tag{6}$$

$$\delta = \arctan\left(\frac{H_0}{2F}\right) \tag{7}$$

Here, the pixel focal length F should be replaced. After obtaining the side opening angle θ and heading opening angle δ , the side offset coverage distance W_1, W_2, W_3, W_4 and pitch coverage distance H_1, H_2 can be calculated because of the existence of them, as shown in formula:

$$\begin{aligned} W_1 &= A \tan(\theta - \omega) & W_2 &= A \tan(\theta + \omega) \\ W_3 &= W_1 + H_2 \tan(\alpha) & W_4 &= W_2 + H_2 \tan(\alpha) \\ W_5 &= W_1 - H_2 \tan(\alpha) & W_6 &= W_2 - H_2 \tan(\alpha) \\ H_1 &= A \tan(\delta - \phi) & H_2 &= A \tan(\delta + \phi) \end{aligned} \tag{8}$$

Then the coordinates of the four corners of the picture can be calculated as follows:

$$\begin{aligned} X_1 &= X_0 + H_2 \sin \kappa - W_3 \cos \kappa \\ Y_1 &= Y_0 + H_2 \cos \kappa + W_3 \sin \kappa \\ X_2 &= X_0 + H_2 \sin \kappa + W_4 \cos \kappa \\ Y_2 &= Y_0 + H_2 \cos \kappa - W_4 \sin \kappa \\ X_3 &= X_0 - H_1 \sin \kappa + W_6 \cos \kappa \\ Y_3 &= Y_0 - H_1 \cos \kappa - W_6 \sin \kappa \\ X_4 &= X_0 - H_1 \sin \kappa - W_5 \cos \kappa \\ Y_4 &= Y_0 - H_1 \cos \kappa + W_5 \sin \kappa \end{aligned} \tag{9}$$

3.3 Calculation of Coordinate Points of Arbitrary Image Position'

Based on the previous calculation basis, the pixel coordinates of other positions are further calculated. The pixel coordinate system of the image is defined in Fig. 4.

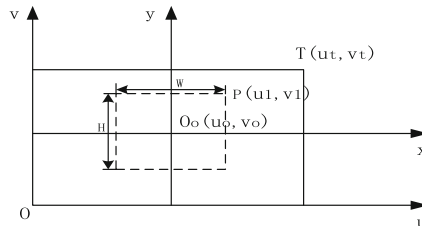


Fig. 4. Image coordinate system

The U-V coordinate system is the image coordinate system, X-Y is the physical coordinate system, where the pixel coordinates of point o are (0, 0), and the pixel coordinates of point t are the width and height of the image, then the pixel coordinates of the center point of the image can be written as follows:

$$\begin{aligned} u_o &= u_t/2 \\ v_o &= v_t/2 \end{aligned} \quad (10)$$

If the pixel coordinates of the target point are known to be

$$\begin{aligned} W &= 2(u_1 - u_0) \\ H &= 2(v_1 - v_0) \end{aligned} \quad (11)$$

Where W is the width of the map with point P as the upper right corner and H is the height of the map. The width and height of the map with the target point as the upper right corner are known, and the heading opening angle and side opening angle are recalculated.

4 Experimental Results

Experiment 1: In order to verify the effectiveness of this method, a set of UAV' POS data is used, as shown in Table 1. From the POS data file, From the POS data file, the image number, latitude and longitude coordinates, flight altitude and flight attitude parameters of the aerial photography equipment can be obtained. In the experiment, The fixed-wing UAV is the main platform for data collection. The focal length of the camera is 30 mm. The experimental platform is Visual Studio, and the programming language is C++.

Table 1. POS data from the UAV

Num	Longitude/	Latitude/	Height/m	Heading angle	Pitch angle	Sideslip angle
1401	110.105758	34.5864911	1795	238.2	0.890842	0.547009
1402	110.105665	34.5865321	1794	238	0.87	0.6234
1403	110.105535	34.586621	1795	237.9	0.85	0.63211
1404	110.105400	34.586755	1795	236.5	0.89	0.67
1405	110.105332	34.586822	1794	237.2	0.8	0.652

The No. 1401 of aerial photo is shown in Fig. 5, with the width and height of 1920 * 1080.

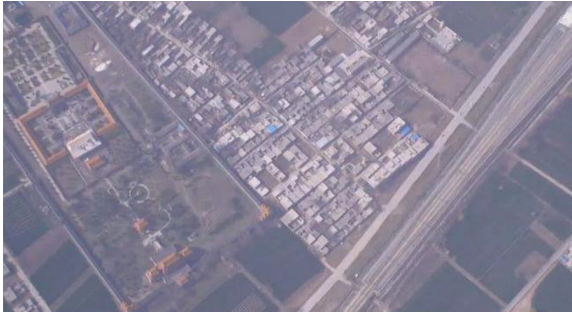


Fig. 5. NO.1401

The corresponding POS data are longitude: 110.105758, latitude: 34.5864911, altitude: 1975, heading angle: 238.2, pitch angle: 0.890842, sideslip angle: 0.547009. Combined with camera parameters, the experimental results are shown in Table 2.

Table 2. The center point and angle coordinates

Position	longitude/°	latitude/°	Longitude (d/m/s)	Latitude (de/m/s)
Central Point	110.1006027	14 point, bold	110 deg 6 min 2 s	34 deg 34 min 59 s
Upper left	110.1019245	12 point, bold	110 deg 6 min 7 s	34 deg 34 min 42 s
Upper right	110.0950483	10 point, bold	110 deg 5 min 42 s	34 deg 35 min 6 s
Bottom left	110.1044152	10 point, bold	110 deg 5 min 42 s	34 deg 34 min 55 s
Bottom right	110.0998345	10 point, italic	110 deg 5 min 59 s	34 deg 35 min 11 s

Experiment 2: On the basis of Experiment 1, the GPS coordinates of other targets in UAV aerial images were further calculated, and the pixel coordinates of calculated targets (9601080), (960,0), (0540), (1920560) were input. The experimental picture was No. 1401, and the UAV POS data were longitude: 110.105758, latitude: 34.5864911, Altitude: 1975, heading angle: 238.2, pitch angle: 0.890842, side angle: 0.547009, combined with pixel focal length and map width height, the experimental results are shown in Table 3. By inputting the pixel coordinates of targets, the method in this paper can calculate the GPS positioning of multiple target points in real time.

Through the calculation of GPS data, the position information can be written into the image, and the image format is converted into KMZ/KML format through global map. Then the image is imported into Google Earth. The coordinates of nine points calculated in Experiment 1 and are shown in Fig. 6.

Table 3. The center point and angle coordinates

Parget pixel coordinates	Longitude/°	Latitude/°	Longitude (d/m/s)	Latitude (de/m/s)
(960, 1080)	110.1006027	14 point, bold	110 deg 6 min 2 s	34 deg 34 min 59 s
(960, 0)	110.1019245	12 point, bold	110 deg 6min 7 s	34 deg 34 min 42 s
(0, 540)	110.0950483	10 point, bold	110 deg 5min 42 s	34 deg 35 min 6 s
(1920, 560)	110.0998345	10 point, italic	110 deg 5min 59 s	34 deg 35 min 11 s



Fig. 6. Nine points on Google Earth

The calculated GPS position information is imported into Google Earth after gradient correction. The results are shown in Fig. 7.

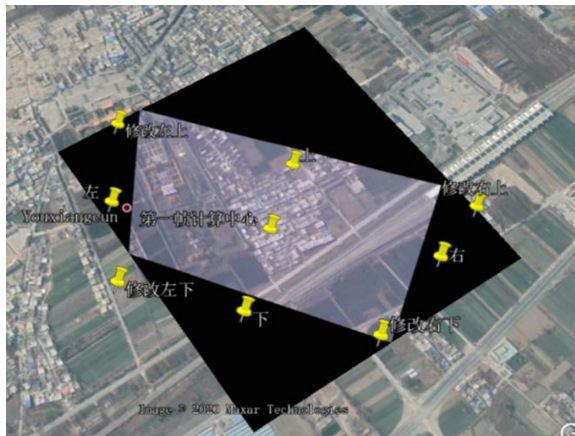


Fig. 7. Importing the original image into Google Earth

It can be seen from the experimental results that this paper realizes the GPS calculation of the target points in aerial images based on POS data. Input the corresponding UAV POS data, combined with the parameters of UAV payload camera, can calculate the plane coordinates of the image center point and the coordinates of four corners in real time. In the further experiments, the other target points are calculated based on the calculation of the center point and four corner points. And the calculation results are written into the image data and compared on Google Earth, which can meet the accuracy requirements.

5 Inclusion

In this paper, the GPS coordinates of sensitive elements and illegal buildings in the project route are calculated in the process of environmental intelligent exploration. First, by obtaining the POS data of the aerial image, the pseudo center coordinates of the image center point under Gaussian projection are calculated, and the position coordinates of the image center point are further calculated through heading correction. Then calculate the GPS positions of the four corners according to the width and height of the image, and further calculate the GPS positions of other target points. Then write the data into the image and compare the accuracy on Google earth. The experimental results show that the GPS position of the target point in the aerial image can be accurately settled based on the pixel coordinates of the POS data and the target point.

Acknowledgements. This work was supported by National Nature Science Foundation of China (NSFC) under Grants No. 61671356, Science and Technology Program of Shaanxi Province under Grants No. 2020GY-136 and 2019ZDLGY14-02-03.

References

1. Tian-Fu, L.I., Cong-Gao, Y.: Brief talk on how to implement the connection between environmental impact assessment system and discharge permit system in EIA technical assessment. *Environm. Sci. Surv.* (2019). Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, p. 13. Springer, Heidelberg (2016)
2. Aung, T.S., Fischer, T.B., Shengji, L.: Evaluating environmental impact assessment (EIA) in the countries along the belt and road initiatives: system effectiveness and the compatibility with the Chinese EIA. *Environ. Impact Assess. Rev.* **81**, 106361 (2020)
3. Liu, X.: Thoughts on the construction of environmental impact assessment system under the new situation. *Environ. Dev.* (2019)
4. Yu, X., Cunkuan, B.: Innovation path of strategic environmental assessment (SEA) for urban and rural planning based on “planning-consciousness & SEA-response”. *urban and rural planning.* (002), 70–81, 95 (2019)
5. No authors listed. Interpretive reporting to improve the effectiveness of clinical laboratory test results. *An ECRI technology assessment. J. Health Care Technol.* **2**(4), 269 (1986)
6. Draga, S.: Problems and solutions in conducting research. *Tech. Commun.* **66**(1), 1–5 (2019)

7. Peng, L.I.: Discussion on safety management of construction site of electric power construction under new situation and its strategy. *Sci. Technol. Vis.* (2018)
8. Xia, H.: The role and problems of environmental impact assessment in governing hydro-power projects in Cambodia. *Beijing Law Rev.* **11**(2), 501–518 (2020)
9. Liu, H, Guo, P., Jin, X.: Indicator system for environmental impact assessment of water resources protection and utilization planning. *IOP Conf. Ser. Earth Environ. Sci.* **514**, 032049 (2020)
10. Strezov, V., Cho, H.H.: Environmental impact assessment from direct emissions of Australian thermal power generation technologies. *J. Clean. Prod.* **270**, 122515 (2020)
11. Chang, J., Wang, Y.D., Yan-Ni, J.I.: Research on image translation and image quality evaluation based on generative adversarial networks (2020)
12. Shichao, G., Dandan, G., Qiongyu, Z., et al.: Research progress of anti-jamming technology of unmanned aerial vehicle (UAV) data link. *IOP Conf. Ser. Earth Environ. Sci.* **816**, 012011 (2020)
13. Yin, H., Du, H., Hao, Q.: High-precision attitude measurement method based on INS/GPS/CNS combined system. *Zhongguo Guanxing Jishu Xuebao/J. Chin. Inertial Technol.* **27**(1), 136–140 (2019)



Encryption and Decryption in Conic Curves Cryptosystem Over Finite Field $GF(2^n)$ Using Tile Self-assembly

Yongnan Li^(✉)

School of National Security, People's Public Security University of China,
Beijing 100038, China
liyongnan.buaa@gmail.com

Abstract. This paper proposes how to accomplish encryption and decryption in conic curves cryptosystem over finite field $GF(2^n)$ using tile self-assembly. Two parameters in ciphertext could be obtained by two separate models of point-multiplication, one of which changes the seed configuration with inputs to adapt the demands. The decryption process is fulfilled by a new designed tile assembly model containing three sub-models that respectively perform the operation of point-multiplication, the operation of negative point and the operation of point-addition. Assembly time complexity of the model to decrypt is $\Theta(n^3)$ and the space complexity is $\Theta(n^6)$.

Keywords: Encryption · Decryption · Conic curves cryptosystem · Finite field $GF(2^n)$ · Tile assembly model

1 Introduction

DNA based cryptography expresses as creating new encryption method according to the complex of DNA structure [6, 19]. Besides this branch, accomplishing the classical cryptography based on DNA computing models is a booming field of modern cryptography. Both symmetrical encryption technology [14] and asymmetrical encryption technology [15] were brought to this developing research field. These researches started from Leonard Adleman proposed a famous experiment to solve Hamiltonian path problem in 1994 [1]. After that, DNA computing began to develop into plenty of research areas including cryptography.

The combination of classical encryption algorithm and DNA computing must be considered under computing models [13] which denote mathematical abstractions of DNA computing. The common used DNA computing models contain tile assembly model [2], sticker model [18], splicing model [12], etc. Tile assembly model is proved to simulate Turing machine [16]. For more details about tile self-assembly, please refer to [17]. The operations over finite field $GF(2^n)$ are very easy to be fulfilled for there is no carry bit in it.

Conic curves cryptography is one of the public key cryptography generally used to compute digital signatures and construct security protocols with hashing functions. There are several types of conic curves respectively constructed over ring Z_n [11], finite field F_p [10] and finite field $GF(2^n)$ [4]. The point-operations such as point-doubling, point-addition and point-multiplication over finite field $GF(2^n)$ are introduced in [4]. Designing tile assembly model to accomplish encryption and decryption in conic curves drives the momentum of DNA based cryptography.

This paper discusses how to obtain the two parameters in ciphertext using tile self-assembly and designs a tile assembly model to accomplish decryption based on two models proposed in our previous work [8,9]. All parameters in the assembly process are considered in bits to avoid the matching problem of polynomial parameters [3,5], the length of which might change in the assembly process. Two separate models of point-multiplication get the result of two parameters in encryption by changing the seed configuration containing input variables. The computation tiles contain 11 bits in every side in the model of decryption including three sub-models. The model of point-multiplication and the model of point-addition are added some additional bits to act as two sub-models performing different functions in decryption. A sub-model calculates the negative point of point-multiplication and makes the output and input of the other two sub-models match with each other strictly. To the best of our knowledge, it is the first research about the encryption and decryption in conic curves cryptosystem using tile self-assembly.

The rest of this paper is organized as follows. Section 2 will introduce the encryption process and decryption process of conic curves cryptosystem. The encryption process and decryption process will be analyzed based on tile assembly model in Sect. 3. Section 4 will design a tile assembly model to implement the decryption process. Last section will provide a conclusion of the contributions.

2 Encryption and Decryption in Conic Curves Cryptosystem over Finite Field $GF(2^n)$

This section briefly introduces the encryption and decryption process in conic curves cryptosystem over finite field $GF(2^n)$ [4]. Let $G(g)$ on conic curves be the generator in cryptosystem. The private key of entity U is $d \in [0, ord(G) - 1]$, where $ord(G) \in \{2^n - 1, 2^n + 1\}$, and then the public key is $U(u) = d \cdot G(g)$.

If sending plaintext m to entity U , the process of encryption is such that: (1) encoding plaintext m as the point $M = M(m)$ on conic curves; (2) finding the public key of U , where $U(u)$ on conic curves; (3) generating a random number $k \in [0, ord(G) - 1]$; (4) computing one of the ciphertext point $Y(y) = k \cdot G(g)$; (5) computing another ciphertext point $X(x) = M(m) \oplus (k \cdot U(u))$; (6) sending $(Y(y), X(x))$ to entity U .

While entity U receiving the ciphertext, the process of decryption is such that: (1) computing $d \cdot Y(y) = d(kG(g))$; (2) computing the negative point $-(d \cdot Y(y))$; (3) computing $M(m) = X(x) \oplus (-(d \cdot Y(y)))$; (4) obtaining m .

3 Analysis of the Encryption and Decryption Process

The encryption process of computing $Y(y)$ is only one point-multiplication. $X(x)$ is composed of one point-multiplication and one point-addition. In general, $X(x)$ in ciphertext could be obtained by computing the point-addition after figuring out the point-multiplication. By analyzing the algorithm of point-multiplication [7], the parameter of point-addition could input into the algorithm of point-multiplication as initial value. The concrete computing process is shown in Algorithm 1. Therefore, there only needs to update the seed configuration of the model of point-multiplication [8]. All computation tiles are not needed to be changed. The two parameters in ciphertext could be calculated by two separate operations of point-multiplication.

Algorithm 1. Computing the second parameter $X(x)$ in ciphertext

Input: $k = (k_n, \dots, k_1, k_0)_2, u = \{u_{n-1}, \dots, u_1, u_0\}$ for point $U(u), x = \{x_{n-1}, \dots, x_1, x_0\}$ for point $X(x), q = \{q_{n-1}, \dots, q_1, q_0\}$ for point $Q(q), e = \{e_{n-1}, \dots, e_1, e_0\}$ for point $E(e), c = \{c_{n-1}, \dots, c_1, c_0\}$ for point $C(c), m = \{m_{n-1}, \dots, m_1, m_0\}$ for point $M(m)$.

Output: $x = \{x_{n-1}, \dots, x_1, x_0\}$ for $X(x) = kU(u) \oplus M(m)$.

```

1:  $Q \leftarrow M, X \leftarrow U$ .
2: for  $i$  from 0 to  $n$  do
3:   if  $k_i = 1$ , then
4:      $C \leftarrow Q \oplus X, E \leftarrow X \oplus X$ .
5:   else, then
6:      $C \leftarrow Q, E \leftarrow X \oplus X$ .
7:    $X \leftarrow E$ .
8:    $Q \leftarrow C$ .
9: end for
10:  $X \leftarrow Q$ .
11: Return  $\{x_{n-1}, \dots, x_1, x_0\}$ .

```

Compared with the algorithm of point-multiplication [8], it only use $M(m)$ replacing $O(\infty)$ to assign the initial value of the parameter in point-multiplication. No changes are needed for the boundary tiles in the vertical line located on the position $(-1, j)$, where $j \geq 0$, in the model of point-multiplication [8]. Only the 6th bits coded as # to represent the initial value of point-multiplication need to be updated in the boundary tiles in the horizontal line located on the position $(i, -1)$, where $i \geq 0$. The 6th bits in the first $n - 1$ boundary tiles located in $(i, -1)$ are assigned as m , the plaintext mapping in finite field $GF(2^n)$. The 6th bits are 0 in the other $n^3 + n^2 - 2n - 2$ boundary tiles in the horizontal line. As shown in Fig. 1, the boundary tiles in $(i, -1)$ of the model of point-multiplication [8] is $\alpha_{00uvv\#xyz\#} = \langle 00uvv\#xyz\#, null, null, null \rangle$, where $u, v, x, y, z \in \{0, 1\}$. This type of boundary tile has to be updated as $\alpha_{00uvvwxyz\#} = \langle 00uvvwxyz\#, null, null, null \rangle$, where $u, v, w, x, y, z \in \{0, 1\}$. Figure 2 shows the re-designed boundary tiles in the horizontal line of seed configuration. Therefore, it adds 32 encoding ways of boundary tiles to compute $X(x)$ compared with the model of point-multiplication [8].

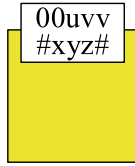


Fig. 1. The boundary tiles in the horizontal line of the seed configuration for computing point-multiplication [8]. The total type number is 32.

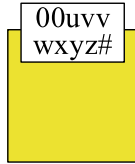


Fig. 2. The boundary tiles in the horizontal line of the seed configuration to calculate $X(x)$ in encryption process. The total type number is 64.

The first step of decryption is only one point-multiplication. It needs to combine one point-addition and one operation of negative point in the second step and the third step to calculate the plaintext after the first step. The point-addition could only be calculated after obtaining the *negative* point of point-multiplication. A new DNA computing model is needed to fulfill the decryption in conic curves cryptosystem over finite field $GF(2^n)$ using tile self-assembly. The following notes focus on designing this new model.

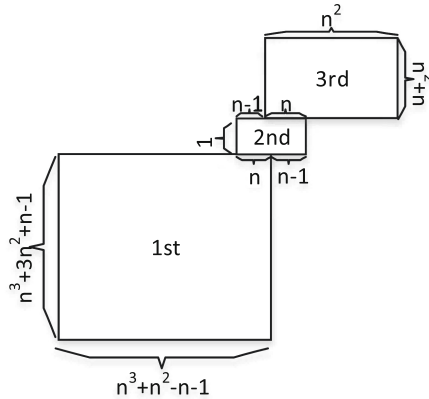


Fig. 3. The position relationship of three sub-models in the final configuration of the whole model.

4 Model of Decryption in Conic Curves Cryptosystem over Finite Field $\text{GF}(2^n)$

In the process of decryption, the computing order is point-multiplication, the *negative* point and the point-addition, one after the other. Three sub-models could accomplish these functions. The first one obtains the result of point-multiplication and transfers the parameter of point-addition. The second one gets the negative point of the result of point-multiplication and the third one fulfills the operation of point-addition. The whole architecture of this model is shown in Fig. 3. The horizontal direction contains $n^3 + 2n^2 - n - 2$ computation tiles and there are $n^3 + 4n^2 + 2n - 2$ computation tiles in the vertical direction. Therefore, the assembly time complexity of this model is $2n^3 + 6n^2 + n - 5$ and the space complexity is $n^6 + 6n^5 + 9n^4 - 4n^3 - 14n^2 - 2n + 4$.

4.1 Sub-model to Compute Point-Multiplication and Transfer Parameter

In the first sub-model, there is no change for all original bit encoding ways and instructions in the computation tiles to compute point-multiplication. Only a bit in every side of computation tile needs to be added to transfer the parameter y in point-addition.

The main function of point-multiplication is achieved by $n + 1$ sub-models and every sub-model contains three parts as shown in Fig. 4. k_i , f' , a , h , p , q , e and c respectively denote the bit in coefficient k of point-multiplication, the modulus number f of $\text{GF}(2^n)$ without the highest bit, the constant number in conic curves $C_{2^n}(a, b)$, constant 1 in divisor of point-addition, the first point parameter, the second point parameter, the result of point-doubling and the result of point-addition.

There are 10 bits in every side of the computation tiles in the model of point-multiplication [8]. As shown in Fig. 5, a more bit could be added to 11th bit to transfer the parameter without changing the other bits. In Fig. 4, A part and C part calculate point-addition and point-doubling. And the invariant parameters such as f' , a and h in the computation tiles located on (i, j) in them transfer to the computation tiles located on $(i+1, j+1)$. B part only makes the 6th bits representing result of point-addition move right-shift $n - 1$ computation tiles and other bits in B part are passed from S side to N side directly.

As shown in Figs. 6 and 7, the 11th bits perform different instructions in three parts. In A part and C part, the 11 bits representing y in the computation tiles located on (i, j) transfer to the computation tiles located on $(i+1, j+1)$ by acting $N_{11} = W_{11}$ and $E_{11} = S_{11}$. The instructions in the computation of B part are $N_{11} = S_{11}$ and the 11th bits in W side and E side are encoded as $\#$ to void the bits.

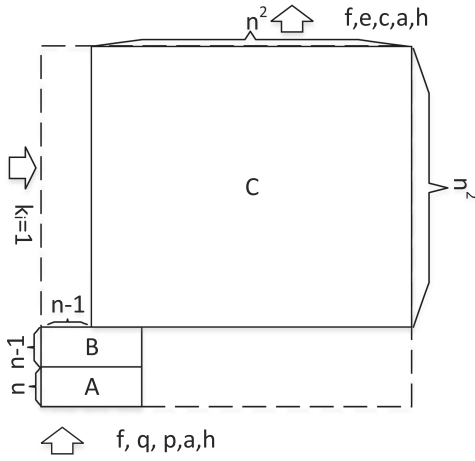


Fig. 4. The structure of sub-model in the model of point-multiplication [8].

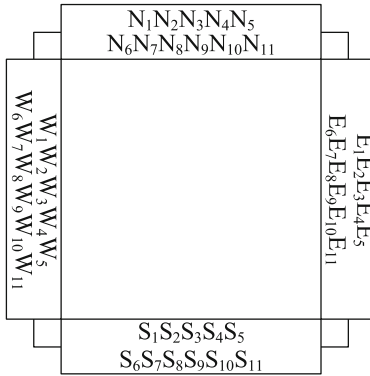


Fig. 5. The tile template.

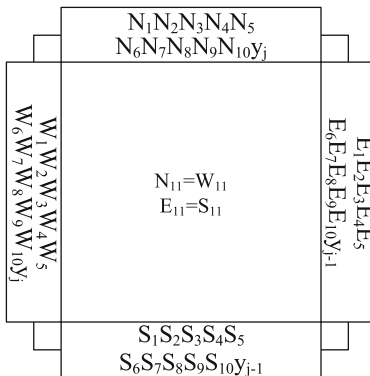


Fig. 6. The computation tile in A part and C part. Subscript j varies from $n - 1$ to 1.

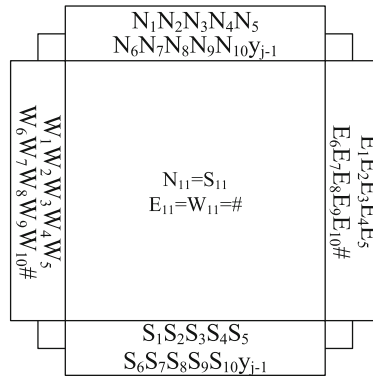


Fig. 7. The computation tile in B part.

4.2 Sub-model to Compute Point-Addition

The computation tiles in the model of point-addition needs 6 bits in every side [9]. In order to match the computation tiles in the sub-model to calculate point-multiplication and transfer parameter, it needs to add 5 more bits in every side of computation tiles in the model of point-addition. The first 6 bits remain unchanged and the other 5 bits are encoded as # to denote invalid bit. Figure 8 demonstrates the computation tiles to compute point-addition in the third sub-model.

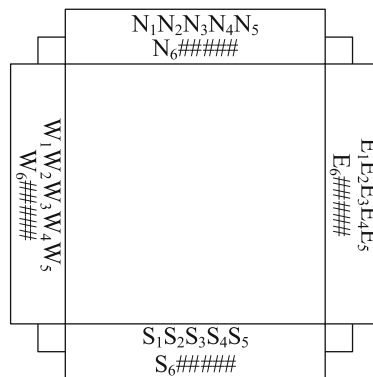


Fig. 8. The computation tiles updated from the model of point-addition [9] in the second sub-model.

4.3 Sub-model to Compute Negative Point

For *negative* point on conic curves over finite field $GF(2^n)$, the definition is $\forall P(t) \in C_{2^n}(a, b)$, $-P(t) = P(t + 1)$, $-P(\infty) = P(\infty)$. There only needs one assembly row to

compute the *negative* point. The output bits have to match the bit order of computation tiles to calculate point-addition.

In the first sub-model, the computation tiles with the 11th bits including valid parameters are shown in Fig. 9. Figure 10 shows the computation tiles with the additional 5 bits to compute point-addition. For the computation tiles to compute negative point, the *S* side and *N* side are respectively map the *N* side of computation tiles in Fig. 9 and the *S* side of computation tiles in Fig. 10.

Figures 11, 12 and 13 show all types of encoding way of the computation tiles in the sub-model to compute *negative* point. The pink tile considers $q \neq \infty$ and $q = \infty$ is covered by brown tile. The computation tiles in Figs. 12(a), 12(b) and 12(c) considering valid input bits and output bits. The gray tile in Fig. 11 is used to pad empty positions and identifies the row *ID* in the *W* side of computation tiles containing valid bits. The pink tile in Fig. 12(d) encoding all bits carrying parameters as 0 pads the empty positions in the *E* side of computation tiles in Figs. 12(a), 12(b) and 12(c). If $q = \infty$, the brown tiles in Figs. 13(a) and 13(b) replace the pink tiles in Figs. 12(a) and 12(b) to transfer q .

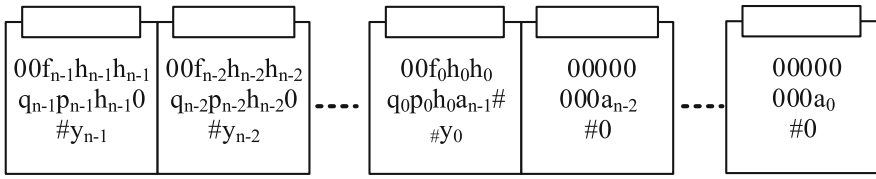


Fig. 9. The computation tiles containing valid output bits in the top assembly row of the first sub-model.

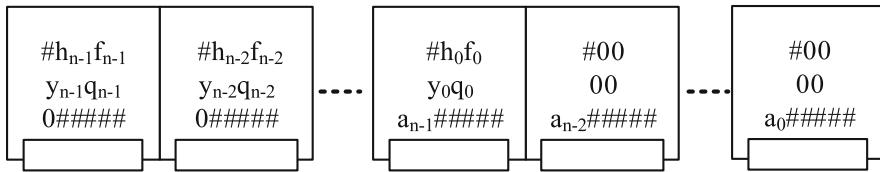


Fig. 10. The computation tiles including valid input bits in the bottom assembly row to compute point-addition in the third sub-model.

4.4 Seed Configuration of the Model of Decryption

Compared with the boundary tiles in the horizontal line of seed configuration in the model of point-multiplication [8], the corresponding boundary tiles in this model add the 11_{th} bits representing the point parameter *y* that will be used in the third sub-model. The boundary tiles in $(-1, j)$, where $j > 0$, encode their 11_{th} bits as #. The sub-model to compute negative point only contains one assembly row and its boundary tile will be in the position $(-1, n^3 + 3n^2 + n - 2)$ and padded by $\beta_{#####0} = \langle null, null, null,$

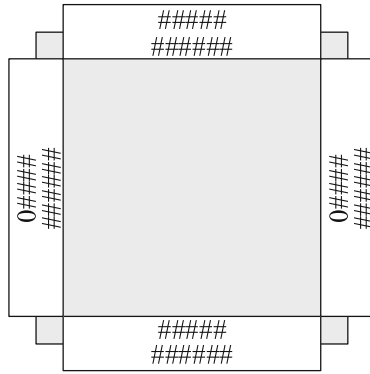


Fig. 11. Gray computation tiles pad empty positions for computing negative point.

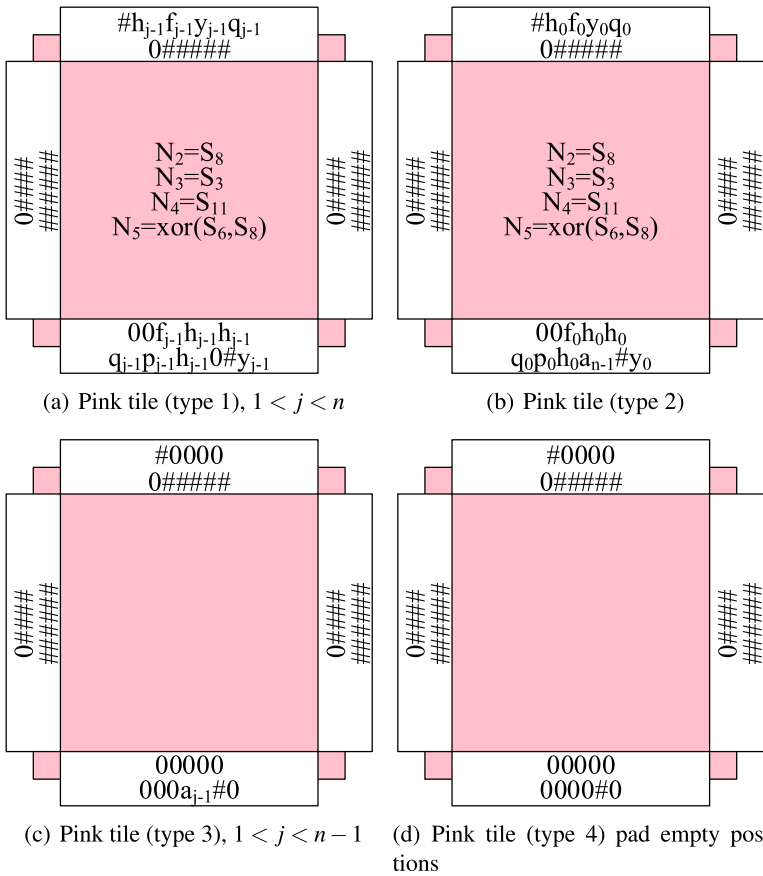


Fig. 12. Pink computation tiles for computing negative point.

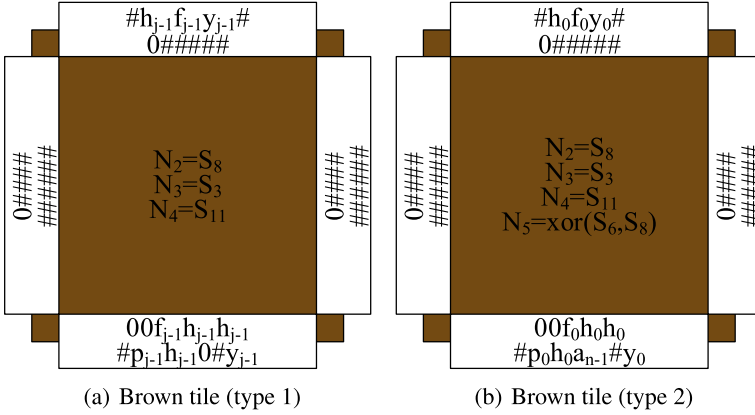


Fig. 13. Brown computation tiles for computing negative point.

$\#\#\#\#\#\#0\rangle$. Another $n^n + n$ assembly rows are needed for the third sub-model to compute point-addition. $\beta_{\#\#\#\#\#\#}$ could be shared by some assembly rows in the first sub-model and the third sub-model.

Let $\Gamma = \{ \alpha_{00uvv\#xyz\#w} = \langle 00uvv\#xyz\#w, null, null, null \rangle, \beta_{\#\#\#\#\#k_m\#\#0\#} = \langle null, null, null, \#\#\#\#\#1\#\#0 \rangle, \beta_{\#\#\#\#\#k_m\#\#1\#} = \langle null, null, null, \#\#\#\#\#1\#\#1 \rangle, \beta_{\#\#\#\#\#k_m\#\#\#} = \langle null, null, null, \#\#\#\#\#1\#\#\# \rangle, \beta_{\#\#\#\#\#00\#} = \langle null, null, null, \#\#\#\#\#00\#\#\#\# \rangle, \beta_{\#\#\#\#\#00\#\#\#\#} = \langle null, null, null, \#\#\#\#\#00\#\#\#\#\# \rangle, \beta_{\#\#\#1\#\#\#\#\#} = \langle null, null, null, \#\#\#1\#\#\#\#\#\# \rangle, \beta_{\#\#\#\#\#\#0} = \langle null, null, null, \#\#\#\#\#\#\#\#\#0 \rangle, \beta_{00\#\#\#\#\#\#\#} = \langle null, null, null, 00\#\#\#\#\#\#\#\#\#\# \rangle, \beta_{1\#\#\#\#\#\#\#\#} = \langle null, null, null, 1\#\#\#\#\#\#\#\#\#\# \rangle, \beta_{11\#\#\#\#\#\#\#\#} = \langle null, null, null, 11\#\#\#\#\#\#\#\#\#\# \rangle, \beta_{\#\#\#\#00\#\#\#\#\#} = \langle null, null, null, \#\#\#\#\#00\#\#\#\#\#\# \rangle, \beta_{\#\#\#\#1\#\#\#\#\#\#} = \langle null, null, null, \#\#\#\#\#1\#\#\#\#\#\# \rangle, \beta_{\#\#\#\#\#\#\#\#\#\#\#} = \langle null, null, null, \#\#\#\#\#\#\#\#\#\#\#\# \rangle \}$ be the set of all boundary tiles, where $u, v, x, y, z, w \in \{0, 1\}$, the number of which is 80. The seed configuration of the model to compute plaintext $M(m)$ is such that

- $\forall i \in \{0, \dots, n-2\}, S(i, -1) = \alpha_{00f_{n-1-i}s_{n-1-i}s_{n-1-i}\#p_{n-1-i}00\#y_{n-1-i}}$,
- $S(n-1, -1) = \alpha_{00f_0s_0s_0\#p_0h_0a_{n-1}\#y_0}$,
- $\forall i \in \{n, \dots, 2n-2\}, S(i, -1) = \alpha_{000000000a_{2n-2-i}\#0}$,
- $\forall i \in \{2n-1, \dots, n^3 + 2n^2 - 2n - 2 + 1\}$,
- $S(i, -1) = \alpha_{000000000\#0}$,
- $\forall j \in \{0, \dots, n^3 + 4n^2 + 2n - 2\}, S(-1, j) =$
 - $\beta_{\#\#\#\#\#k_m\#\#0\#}$, if $j < n^3 + 3n^2 + n - 1$ and $j \bmod (n^2 + 2n - 1) = 0$,
 - $\beta_{\#\#\#\#\#k_m\#\#1\#}$, if $j < n^3 + 3n^2 + n - 1$ and $0 < \bmod(n^2 + 2n - 1) < n - 1$,
 - $\beta_{\#\#\#\#\#k_m\#\#\#}$, if $j < n^3 + 3n^2 + n - 1$ and $j \bmod (n^2 + 2n - 1) = n - 1$,
 - $\beta_{\#\#\#\#\#\#00\#}$, if $j < n^3 + 3n^2 + n - 1$ and $n - 1 < j \bmod (n^2 + 2n - 1) < 2n - 1$,
 - $\beta_{\#\#\#\#00\#\#\#\#}$, if $j < n^3 + 3n^2 + n - 1$ and $j \bmod (n^2 + 2n - 1) = 3n - 2$,
 - $\beta_{\#\#\#1\#\#\#\#\#}$, if $j < n^3 + 3n^2 + n - 1$ and $j \bmod (n^2 + 2n - 1) = S \cdot n - 2$ and $S > 3$ is positive integer,
 - $\beta_{\#\#\#\#\#\#\#\#\#\#\#0}$, if $j = n^3 + 3n^2 + n - 1$,

- $\beta_{00\#\#\#\#\#\#\#}$, if $j = n^3 + 3n^2 + n - 2$,
- $\beta_{1\#\#\#\#\#\#\#}$, if $j > n^3 + 3n^2 + n - 2$ and $0 < (j + 2 - n^3 - 3n^2 - n) < n - 1$,
- $\beta_{11\#\#\#\#\#\#\#}$, if $j > n^3 + 3n^2 + n - 2$ and $(j + 2 - n^3 - 3n^2 - n) = n - 1$,
- $\beta_{\#\#\#\#00\#\#\#\#}$, if $j > n^3 + 3n^2 + n - 2$ and $(j + 2 - n^3 - 3n^2 - n) = 2n - 1$,
- $\beta_{\#\#\#\#1\#\#\#\#\#}$, if $j > n^3 + 3n^2 + 3n - 3$ and $(j + 3 - n^3 - 3n^2 - n) \bmod n = 0$,
- $\beta_{\#\#\#\#\#\#\#\#\#\#\#}$, other cases.

5 Conclusions

This paper discusses how to implement encryption and decryption in conic curves cryptosystem over finite field $\text{GF}(2^n)$ using tile self-assembly. There are two parameters in the ciphertext. One of them is the result of an operation of point-multiplication and the other one is a combination of point-addition and point-multiplication. Both of them could be figured out by the model of point-multiplication [8], and computing second parameter needs to improve the seed configuration of point-multiplication without changing the computation tiles. The decryption process is accomplished by a new designed tile assembly model including three sub-models. The first sub-model calculates point-multiplication. Point-addition is performed by the third sub-model. The two sub-models are updated from the model of point-multiplication [8] and the model of point-addition [9] proposed in our previous works. The second sub-model obtains the negative point of the result of point-multiplication and makes the output of the first sub-model and the input of the third sub-model match each other. The assembly time complexity of the model to decrypt is $2n^3 + 6n^2 + n - 5$ and the space complexity is $n^6 + 6n^5 + 9n^4 - 4n^3 - 14n^2 - 2n + 4$.

Acknowledgment. This study is sponsored by the National Natural Science Foundation of China under Grant No. 61702523 and the Fundamental Research Funds for the Central Universities under Grant No. 2020JKF303.

References

1. Adleman, L.M.: Molecular computation of solutions to combinatorial problems. *Science* **266**, 1021–1024 (1994)
2. Amarioarei, A., et al.: One dimensional DNA tiles self assembly model simulation. *Int. J. Unconv. Comput.* **13** (2018)
3. Barua, R., Das, S.: Finite field arithmetic using self-assembly of DNA tilings. In: *Proceeding of the 2003 Congress on Evolutionary Computation (CEC 2003)*, vol. 4, pp. 2529–2536. IEEE (2003)
4. Cai, Y.Q., Lei, Z., Jin, Y.Y.: A public-key cryptosystem based on conic curve in finite field $\text{GF}(2n)$. *Acta Electronica Sinica* **34**(8), 1464–1468 (2006)
5. Cheng, Z.: Computation of multiplicative inversion and division in $\text{GF}(2(n))$ by self-assembly of DNA tiles. *J. Comput. Theor. Nanosci.* **9**(3), 336–346 (2012)
6. Gehani, A., LaBean, T., Reif, J.: DNA-based cryptography. In: Jonoska, N., Păun, G., Rozenberg, G. (eds.) *Aspects of Molecular Computing*. LNCS, vol. 2950, pp. 167–188. Springer, Heidelberg (2003). https://doi.org/10.1007/978-3-540-24635-0_12
7. Hankerson, D., Menezes, A., Vanstone, S.: *Guide to elliptic curve cryptography*, pp. 96–97. Springer, Heidelberg (2004)

8. Li, Y.: A tile assembly model to calculate point-multiplication on conic curves over finite field $\text{GF}(2^n)$. In: Proceeding of 18th IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA 2020), Exeter, UK. IEEE, December 2020 (in press)
9. Li, Y., Xiao, L.: Arithmetic computation using self-assembly of DNA tiles: point-addition on conic curves over finite field $\text{GF}(2^n)$. In: Proceeding of 25th International Conference on Parallel and Distributed Systems (IPCADS 2019), Tianjin, China, pp. 888–895. IEEE, December 2019
10. Li, Y., Xiao, L., Hu, Y., Liang, A., Tian, L.: Parallel algorithms for cryptosystem on conic curves over finite field fp . In: Proceedings of 2010 9th International Conference on Grid and Cooperative Computing (GCC), pp. 163–167. IEEE (2010)
11. Lin, S., Wang, B., Li, Z.: Digital multisignature on the generalized conic curve over Zn . *Comput. Secur.* **28**(1–2), 100–104 (2009)
12. Pan, L., Song, B., Nagar, A.K., Subramanian, K.: Language generating alphabetic flat splicing P systems. *Theoret. Comput. Sci.* **724**, 28–34 (2018)
13. Paun, G., Rozenberg, G., Salomaa, A.: *DNA Computing: New Computing Paradigms*. Springer, Heidelberg (2005)
14. Rakheja, P.: Integrating DNA computing in international data encryption algorithm (IDEA). *Int. J. Comput. Appl.* **26**(3), 1–6 (2011)
15. Tanaka, K., Okamoto, A., Saito, I.: Public-key system using DNA as a one-way function for key distribution. *Biosystems* **81**(1), 25–29 (2005)
16. Winfree, E., Yang, X., Seeman, N.C.: Universal computation via self-assembly of DNA: some theory and experiments. *DNA Based Comput. II* **II**(44), 191–213 (1998)
17. Yuriy, B.: Nondeterministic polynomial time factoring in the tile assembly model. *Theor. Comput. Sci.* **395**(1), 3–23 (2008)
18. Zhang, C., Zhu, W., Zhou, Q.: The algorithms of weighting based on DNA sticker model. In: Zhou, Q., Miao, Q., Wang, H., Xie, W., Wang, Y., Lu, Z. (eds.) *ICPCSEE 2018. CCIS*, vol. 902, pp. 250–262. Springer, Singapore (2018). https://doi.org/10.1007/978-981-13-2206-8_22
19. Zkaynak, F., Yavuz, S.: Analysis and improvement of a novel image fusion encryption algorithm based on DNA sequence operation and hyper-chaotic system. *Nonlinear Dyn.* **78**(2), 1311–1320 (2014)



Optimizing Embedding-Related Quantum Annealing Parameters for Reducing Hardware Bias

Aaron Barbosa¹, Elijah Pelofske¹, Georg Hahn²(✉), and Hristo N. Djidjev¹

¹ CCS-3 Information Sciences, Los Alamos National Laboratory,
Los Alamos, NM 87545, USA

{[abarbosa](mailto:abarbosa@lanl.gov),[epelofske](mailto:epelofske@lanl.gov),[djidjev](mailto:djidjev@lanl.gov)}@lanl.gov

² T.H. Chan School of Public Health, Harvard University, Boston, MA 02115, USA
ghahn@hsph.harvard.edu

Abstract. Quantum annealers have been designed to propose near-optimal solutions to NP-hard optimization problems. However, the accuracy of current annealers such as the ones of D-Wave Systems, Inc., is limited by environmental noise and hardware biases. One way to deal with these imperfections and to improve the quality of the annealing results is to apply a variety of pre-processing techniques such as spin reversal (SR), anneal offsets (AO), or chain weights (CW). Maximizing the effectiveness of these techniques involves performing optimizations over a large number of parameters, which would be too costly if needed to be done for each new problem instance. In this work, we show that the aforementioned parameter optimization can be done for an entire class of problems, given each instance uses a previously chosen fixed embedding. Specifically, in the training phase, we fix an embedding E of a complete graph onto the hardware of the annealer, and then run an optimization algorithm to tune the following set of parameter values: the set of bits to be flipped for SR, the specific qubit offsets for AO, and the distribution of chain weights, optimized over a set of training graphs randomly chosen from that class, where the graphs are embedded onto the hardware using E . In the testing phase, we estimate how well the parameters computed during the training phase work on a random selection of other graphs from that class. We investigate graph instances of varying densities for the Maximum Clique, Maximum Cut, and Graph Partitioning problems. Our results indicate that, compared to their default behavior, substantial improvements of the annealing results can be achieved by using the optimized parameters for SR, AO, and CW.

1 Introduction

Quantum annealers such as the ones designed by D-Wave Systems, Inc. [3] are able to find approximate solutions to NP-hard problems of very high quality by resorting to a technique called quantum annealing. To be precise, the D-Wave

annealer is designed to solve optimization problems requiring the minimization of a function of the form

$$H(x_1, \dots, x_n) = \sum_{i=1}^n h_i x_i + \sum_{i < j} J_{ij} x_i x_j, \quad (1)$$

where the linear weights $h_i \in \mathbb{R}$ and the quadratic couplers $J_{ij} \in \mathbb{R}$ are specified by the user and define the problem, and x_i are unknown binary variables, where $i, j \in \{1, \dots, n\}$. To minimize Eq. (1), the D-Wave annealer maps the connectivity of the logical qubits in Eq. (1), i.e., the graph defined by the set of edges (i, j) for which $J_{ij} \neq 0$, to the qubits and links between them on its hardware chip, called a *Chimera* graph (see [2] for a graphical representation of the Chimera graph). The process of submitting a problem to a D-Wave machine is as follows:

1. The problem of interest must be represented as the minimization of a function of the form of Eq. (1). The function of Eq. (1) is called a *QUBO* (quadratic unconstrained binary optimization) problem if $x_i \in \{0, 1\}$ for $i \in \{1, \dots, n\}$, and an *Ising* problem if $x_i \in \{-1, +1\}$ for $i \in \{1, \dots, n\}$. Both QUBO and Ising formulations are equivalent [2]. We can represent the function of Eq. (1) as a graph P itself having n vertices, one for each variable x_i , $i \in \{1, \dots, n\}$. In this representation, each vertex i is assigned a vertex weight h_i , and each edge between vertices i and j is assigned the edge weight J_{ij} .
2. Next, the problem graph P is mapped onto the hardware of the D-Wave 2000Q annealer. Since it is usually not the case that the structure of the graph P perfectly matches the structure of the Chimera graph of the D-Wave 2000Q, a *minor embedding* of P onto the Chimera graph has to be computed. In such an embedding, some logical qubits in Eq. (1) become a *chain*, which is a set of hardware qubits on the chip linked together in a way that prompts them to take the same value at the end of the anneal. Defining the chains requires the specification of a parameter determining the strength of the coupling between the qubits in a chain (the *chain strength* or *chain weight*). The minor-embedded problem P onto the Chimera graph corresponds to a new graph P' , which is a subgraph of the Chimera graph.
3. At the start of the annealing process, the qubits used in the embedding of P' onto the D-Wave hardware are initialized in an equal superposition [3, 7]. During annealing, the system is slowly driven from the neutral transverse field Hamiltonian to the user-specified QUBO or Ising problem H of Eq. (1) while remaining, in theory, in the ground state.
4. Since all qubits in a chain represent one logical qubit, they act as one in theory. However, this is not guaranteed in practice, and hardware qubits in a chain might not always take the same value after annealing. In this case, we speak of a *broken chain*. There is no unique way to assign a definite value to each logical qubit employed in Eq. (1) based on its broken chain, and D-Wave offers several default methods to *unembed* chains, i.e., to decide on the value to be assigned to broken chains.

In practice, several sources of error potentially decrease the quality of the solution returned by the D-Wave annealer. First, before annealing, the linear weights and quadratic couplers in Eq. (1) have to be mapped to electrical currents on the hardware chip using a linear-to-analog converter [7]. This conversion works with a finite precision of 8 bits, thus necessarily resulting in weights spanning a range larger than 8 bits to be mapped imprecisely due to rounding errors. Moreover, so-called *leakage* may occur on the physical chip from the coupler J_{ij} to the adjacent linear weights h_i and h_j , where $i, j \in \{1, \dots, n\}$. This can likewise alter the linear weights h_i and h_j [6], where the effect is reported to be more serious for chained qubits.

One simple way to mitigate such hardware biases is the so-called *spin reversal* (SR) or *gauge transform*. Spin reversal works on Ising problems and is based on the idea that although, theoretically, quantum annealing is invariant under a gauge transformation (i.e., the reversal of spin-up and spin-down in a quantum system), the D-Wave annealer is not a closed system and thus breaks gauge symmetry. As a consequence, two Ising problems in which certain spins have been flipped result in (slightly) different systems when mapped onto the annealer. Solving several Ising problems with a certain number of spin reversed qubits allows us to average results, and balance out errors. In practice, we select an arbitrary subset of variables $S \subseteq \{1, \dots, n\}$ in an Ising problem, and substitute the corresponding variables as $x_i \rightarrow -x_i$ for all $i \in S$ in the Ising problem (the corresponding linear terms and quadratic couplers have to be modified as well). This is equivalent to re-interpreting an up spin as a down spin and vice versa, thus leaving the ground state of the Ising problem invariant, but having the potential to reduce analog and systematic errors on the device. The spin reversal can be applied on two different levels, either before or after embedding Eq. (1) onto the hardware (see Sect. 2). In this work we explore both variants.

Second, in theory, all qubits evolve simultaneously during the anneal process, experiencing equal changes to the tunneling energy and equally contributing to the classical energy function [1]. In practice, however, qubits freeze out at different times during the anneal [10], which might bias the qubit states at readout after annealing. To this end, D-Wave offers to set *anneal offsets* (AO) for all individual qubits with the aim to improve the solution quality. In order to synchronize the evolution of the qubits, the D-Wave 2000Q device offers the ability to delay or advance the evolution of individual qubits within predefined ranges, meaning that qubits can individually be set to start their anneal process earlier or later compared to the default schedule. We consider setting individual anneal offsets for all qubits in this work. Analogously to spin reversal, we consider applying anneal offsets in two different ways, either using separate AO for each individual qubit, or using the same AO for all qubits in each chain (see Sect. 2).

Third, all couplers on the D-Wave hardware require the specification of a weight, and as such, when embedding a logical qubit as a chain on the D-Wave hardware, couplers have to also be assigned to all pairwise connections of chained qubits. The chain weight is typically set by D-Wave, in which case some overall weight is equally distributed to all couplers in a chain. However, it can also be

set manually. We read out the total chain weight (CW) determined by D-Wave for each chain, and aim to re-distribute it along the chain in an optimal way.

Although SR, AO, and CW can be effective for removing hardware biases in the annealer, it is non-trivial to optimally select the actual qubits to which a spin reversal is applied, or the values of the anneal offsets or chain weights for each new problem instance being solved. This is because each technique has around 2000 degrees of freedom. Previous work has improved upon the spin reversal transform by using classical optimization in order to find an optimal set of qubits to spin reverse [9]. Concerning the anneal offsets, D-Wave reports that longer chains are likely to freeze out sooner during the anneal process due to their lower effective tunneling energy [1], and they recommend delaying the evolution of those qubits which will be subjected to strong magnetic fields relative to the other working qubits. Moreover, [8] suggest to advance qubits in their evolution if their final state does not contribute to the energy of the classical solution.

Since tuning all qubits for an application of SR, AO, or CW individually for each new problem instance under consideration is infeasible, we propose a different approach in this work. We aim to optimize SR, AO and CW with respect to a whole class of input problems. We carry out all optimizations in the classical set-up of training and validation sets for three NP-hard problems, the Maximum Clique, Maximum Cut, and Graph Partitioning problems. Using a differential evolution optimizer, we tune the average performance of the spin reversal transform, anneal offsets, or chain weights across all of the training graphs, and evaluate our optimized sets of parameters on a set of test graphs.

The article is structured as follows. In Sect. 2, we describe the background of SR, AO, and CW, as well as the optimization framework we employed. We also introduce the NP-hard problems we consider (Maximum Clique, Maximum Cut, and Graph Partitioning). Experimental results are reported in Sect. 3. The article concludes with a discussion in Sect. 4.

2 Methods

In this section, we justify our approach of using a fixed embedding for optimizing SR, AO, and CW (Sect. 2.1). We provide more details on the two types of spin reversal we apply (Sect. 2.2), as well as on anneal offsets (Sect. 2.3) and chain weights (Sect. 2.4). Section 2.5 defines the NP-hard problems we consider. A description of the optimization we perform to tune the application of SR, AO, and CW is given in Sect. 2.6.

2.1 Using a Fixed Embedding

Using the same (fixed) embedding is a key ingredient of our approach. If we want the same set of optimized parameters to work for multiple problems, we need at least one invariant, and it is, in this case, the hardware embedding. We use the fact that, for an NP-hard problem, the limiting factor for embedding its problem graph P is the largest complete graph that can be embedded onto the quantum

annealing hardware. Hence, instead of using an arbitrary embedding of a complete graph that the D-Wave's embedding method *minorminer* would randomly find, we can use a fixed one, and optimize the hardware related parameters using that fixed embedding. In addition, since we will use the same embedding many times, it makes sense to choose one with as good properties as possible. For this reason, we try several complete graph embeddings and choose one that gives the best performance overall, i.e., the best QUBO/Ising value when using default D-Wave parameters, separate for each of the three considered problems.

2.2 Spin Reversal

Suppose we are given an Ising problem in the form of Eq. (1) and a set $S \subseteq \{1, \dots, n\}$ of spins to be reversed. To transform a particular x_i from -1 to $+1$, while keeping the value of Eq. (1) unchanged, we define a new function H' with $h'_i \rightarrow -h_i$ and $J'_{ij} \rightarrow -J_{ij}$, $J'_{ji} \rightarrow -J_{ji}$ for all $i \in S$, where $j \in \{1, \dots, n\}$. Note that the ground state energies of H and H' are identical, and each minimum of H' is a minimum of H with the i th variable having a flipped sign. To apply spin reversal to a set S , we apply the above transformation to each $i \in S$.

It is not obvious how the set S should be chosen to maximize the benefit of the spin reversal. As remarked in [6], reversing too few spins leaves the Ising model almost unchanged, whereas applying spin reversal to too many qubits likely results in many pairs of connected qubits being transformed, thus effectively leaving the corresponding quadratic couplers unchanged. In both cases, the spin reversal transform might only have little effect. Hence, the default spin reversal implemented by D-Wave flips roughly half of the qubits randomly.

When optimizing the spin reversal for a particular problem, we are thus asked to determine for each involved qubit a binary spin reversal indicator, denoting if a particular qubit is spin reversed or not.

Note that spin reversal can be applied on two levels: First, we can flip the logical qubits in the formulation of Eq. (1). This will be referred to as *spin reversal on the chain level*, since in this case qubits which are being mapped onto the hardware as chains are either all reversed or all non-reversed. Second, we can embed the original problem graph P (see Sect. 1) and read out the embedded Ising problem in the graph representation P' . The resulting Ising problem likely consists of more qubits, due to some logical qubits being mapped to a set of physical qubits of the D-Wave hardware, and we can flip each hardware qubit individually. This will be referred to as *spin reversal on the qubit level*.

2.3 Anneal Offsets

Typically, all hardware qubits being used in a problem embedded onto the D-Wave quantum chip undergo the same anneal process simultaneously. In such a case, it is normal that qubits freeze out at different times during the anneal [10], which, however, might affect negatively the dynamics of the annealing and prevent the system from reaching its ground state. To this end, in the newest

generation of the annealer, the start of the anneal process of individual qubits can be moved forward or backward in time, within specified limits.

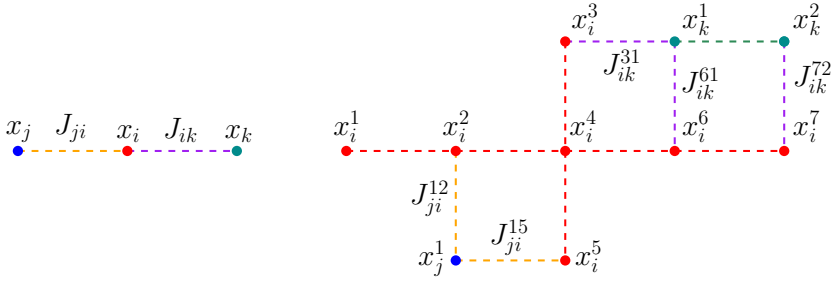


Fig. 1. The logical qubit x_i (left) is mapped onto a chain of seven physical qubits x_i^1, \dots, x_i^7 (right), and the logical couplers J_{ji} and J_{ik} (connecting x_i to qubits x_j and x_k) are mapped to the physical couplers J_{ji}^{15} , J_{ji}^{12} and J_{ik}^{31} , J_{ik}^{61} , J_{ik}^{72} , respectively.

For D-Wave 2000Q, individual anneal offsets can be specified for each hardware qubit using the parameter *anneal_offsets*, where the value 0 denotes no offset, and positive (negative) values indicate that a qubit’s anneal begins ahead of (behind) the standard schedule. The range of the variable *anneal_offsets* is machine dependent, and can be queried with *anneal_offset_ranges*. Moreover, anneal offsets are discrete, with a machine dependent step size specified in *anneal_offset_step*. We tune the anneal offset of each involved qubit with an optimization within the range *anneal_offset_ranges* (given as boundary condition to the optimizer) over a discrete search space, similar to spin reversal optimization.

2.4 Bias Distribution on Physical Qubits

Typically, logical qubits have to be mapped to chains of hardware qubits when computing a minor embedding of a QUBO/Ising problem of Eq. (1) onto the D-Wave Chimera graph. In this case, a logical qubit (variable) x_i is mapped onto a chain $\{x_i^1, \dots, x_i^l\}$ having $l \in \mathbb{N}$ hardware qubits. The linear bias h_i and the quadratic biases J_{ij} have to be distributed between the physical qubits and the links between them (see Fig. 1). The default method of D-Wave distributes h_i and J_{ij} uniformly between the qubits $\{x_i^1, \dots, x_i^l\}$ and the links between the physical qubits (chains) implementing x_i and x_j , respectively. However, there is no evidence that such a method is optimal. In fact, Pudenz [12] has compared the default method with two other distribution strategies and has shown that in some cases the alternative methods work better.

However, none of the previous strategies considers the possible effect of hardware biases, or looks at bias distribution strategies to mitigate such issues. Here we address this problem, using our fixed embedding approach, and tackle the bias distribution problem (i.e., how to distribute the biases on the physical qubits

and couplers in a way as to optimize the annealing results) as an optimization problem. The optimization of linear weights and quadratic couplers is done separately, denoted as CW(L) and CW(Q), respectively. When optimizing linear weights, we evenly distribute the quadratic weights among the quadratic couplers, and analogously for the linear weights when optimizing quadratic couplers.

2.5 Formulations of the NP-hard Problems Studied

We consider three classical NP-hard problems in this work, the Maximum Clique, Maximum Cut, and Graph Partitioning problems. For a graph $G = (V, E)$ with vertex set V and edge set E , a *clique* in G is any subgraph C of G that is *complete*, i.e., there is an edge between each pair of vertices of C . The *Maximum Clique problem* asks us to find a clique of maximum size. A formulation of the Maximum Clique problem in the form of Eq. (1) can be found in [11].

Similarly, a *cut* of the graph is any partition of V into two disjoint sets, that is $V = V_1 \cup V_2$ and $V_1 \cap V_2 = \emptyset$. The *cut size* of any cut is the number of edges having one endpoint in V_1 and one endpoint in V_2 , that is $|\{e = (v, w) : v \in V_1, w \in V_2\}|$. The *Maximum Cut problem* asks us to find a cut of maximum size. A formulation of the Maximum Cut problem as an Ising problem can be found in [5].

Last, the *Graph Partitioning problem* asks us to divide the set of vertices V into two disjoint and balanced sets (partitions) V_1 and V_2 , satisfying $V = V_1 \cup V_2$ and $V_1 \cap V_2 = \emptyset$, such that the size of V_1 and V_2 differs by at most one and the number of *cut edges* $\{e = (v, w) : v \in V_1, w \in V_2\}$ between the two partitions is minimized. An Ising formulation for Graph Partitioning is given in [4].

2.6 Differential Evolution Optimization

In this work, we aim to tune three parameters per qubit, that is, its spin indicator for spin reversal, its anneal offset, and its chain weights in case we are dealing with a chained qubit on the D-Wave Chimera hardware.

To carry out the optimization we employ the *differential optimization* solver of the *SciPy* library in Python [14], available under the command `scipy.optimize.differential_evolution()`, which implements the algorithm of Storn and Price [13].

We employ the differential optimizer with a population size of 80 and 50 generations. We do not use the option *polishing*, i.e., no steepest decent with a quasi-Newton method is performed to fine-tune the solution. All remaining parameters are left at their default values. The initial population is random, but we made sure to include the default parameters for SR, AO, and CW given by D-Wave Systems, Inc. Lastly, we use elitism in the optimization, i.e., we make sure that the best solution is always passed on to the next generation, as opposed to the possibility of being replaced by crossover and random selection operations.

3 Experimental Analysis

We will perform the optimization on a class of random test graphs, separately for the three problems introduced in Sect. 2.5, and evaluate the performance on

a series of (unseen) testing graphs. With this, we aim to find out if it is possible to enhance the performance of the D-Wave 2000Q on a whole class of problem instances, since it is easy to see that due to the large search space, optimizing parameters for each newly solved problem is infeasible.

In all experiments, we fix the anneal duration at $1000\mu\text{s}$. We generate 10 training and 10 testing graphs, each with 65 vertices, the size of the largest complete graph embeddable onto the D-Wave hardware. We vary the density of the training and validation graphs in $\{0.25, 0.50, 0.75\}$. We use the majority vote unembedding algorithm. For Maximum Clique and Maximum Cut we use a chain strength of 1, and for Graph Partitioning a chain strength of $20 \cdot 32 \cdot 33 \cdot d$, where d is the graph density. This is similar to the choice in [4], where the chain strength for Graph Partitioning is set to a prefactor multiplied with an estimate of the value of the objective function.

For the optimization, for each fixed point in the parameter search space, we perform 1000 anneals per graph, and record the average performance across all 10 training graphs, measured in both the value of the QUBO/Ising objective function and the energy (before unembedding) returned by the D-Wave annealer. For testing, we perform 10000 anneals per graph.

When reporting the experimental results, we denote by *Default-RE* the default behavior of the D-Wave annealer with a random embedding and with all other parameters set to their default values. As the optimization is performed on the same embedding, it is reasonable to try to find one that will result in the best performance on average. Hence, we try 30 random embeddings and choose one for each problem that yields the best objective function value during forward annealing with default parameters (since Maximum Clique and Maximum Cut are maximization problems, the higher the value the better, whereas for Graph Partitioning, which is a minimization problem, lower is better). We denote this as *Default-OE* (for default D-Wave with optimized embedding). Moreover, we denote by $\text{SR}(\text{Q})$ and $\text{SR}(\text{C})$ the tuned spin reversal on the qubit or chain level, and similarly $\text{AO}(\text{Q})$ and $\text{AO}(\text{C})$ denote the tuned anneal offsets on the qubit or chain level. Moreover, $\text{CW}(\text{L})$ and $\text{CW}(\text{Q})$ refer to setting the chain weights of linear or quadratic couplers. Since the D-Wave 2000Q features *auto_scale* and *extended_j_range* are mutually exclusive, and because we use auto scaling for all experiments, we optimized $\text{CW}(\text{Q})$ without the extended J range feature.

We assess the performance of all methods using the time-to-solution metric, defined as the time to reach an optimum solution at least once with probability 0.99. It is computed as $\text{TTS} = T_{\text{QPW}} \cdot \log(0.01) / \log(1 - p)$, where T_{QPW} is the solve time on D-Wave, and p is the proportion of times the optimal solution was found. Two caveats are worth mentioning: For problem instances where the optimal solution can be found using a classical solver, we are able to compute the time-to-optimal-solution, which we simply denote by TTS (time-to-solution). In case the optimal solution cannot be found in reasonable time, we relax this metric to *time-to-best-solution*, denoted by TBS, which uses the best solution found by any of the methods, instead of the provably best one. The TBS measure depends on the set of algorithms employed in the study, their parameters, and

the D-Wave samples on which these algorithms are run. However, the setup of the simulations presented here is fixed, thus making the TBS measure well-defined. Lastly, the TTS measure we report does not include any classical computation time, nor the training portion of the optimization done for each problem.

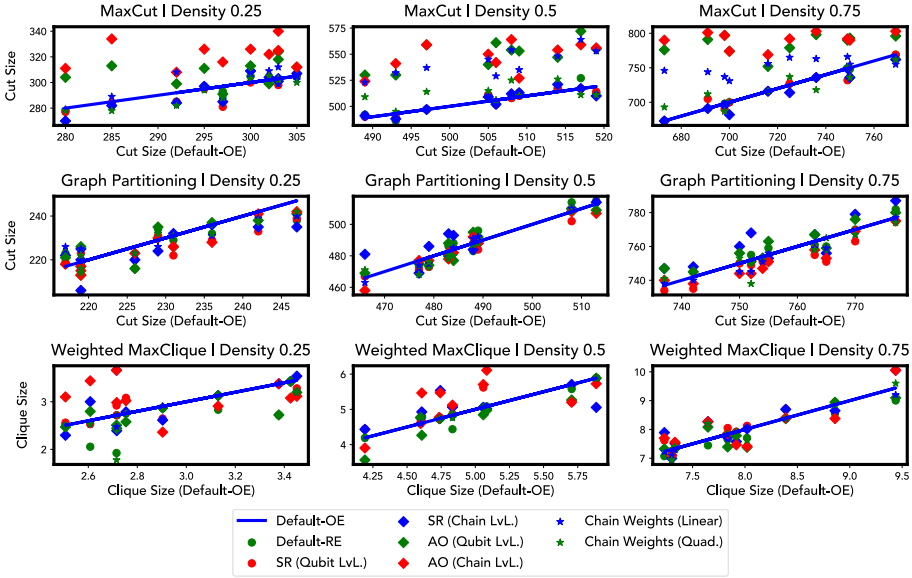


Fig. 2. Performance on test graphs compared to Default-OE for graphs of density 0.25 (left column), 0.5 (middle column), and 0.75 (right column). Metric is the cut size for the Maximum Cut problem (top row), cut size for the Graph Partitioning problem (middle row), and clique size for the Maximum Clique problem (bottom row).

Figure 2 gives a graphical representation of our results on the suite of test graphs for all three problems. We measure results in the raw values of the found cut size (for Maximum Cut and Graph Partitioning) or clique size for the Maximum Clique problem. We observe that, compared to the baseline of Default-OE, SR as well as AO and CW seem to perform well for Maximum Cut. For the other two problems, it is mostly SR and AO on the chain level which perform best.

Next, the results for the TTS estimations are given in Table 1. It is complicated to rank the performances of these techniques, since two measures are of relevance here. First, a low TTS or TBS time is desired for any method. However, some methods might be able to achieve a low TTS/TBS time, but only at the expense of solving fewer test graphs than others, and a method solving almost all graphs usually incurs a higher TTS/TBS time (this occurs, for instance, for the Maximum Cut problem and density 0.25). We thus denote both the measured TTS/TBS time for the graph that could be solved, as well as the number of the graph problems for which the best solution could be found, in parentheses.

Table 1. TBS (white) and TTS (gray) for the test graphs. Number of test graphs for which optimal solutions were found in parentheses. The best TTS and the highest number of optimal/best solutions found for each problem/density combination are given in bold. In case of a tie, the combination with the smallest TTS is chosen.

Problem	Density	Default-OE	Default-RE	SR(Q)	SR(C)	AO(Q)	AO(C)	CW(L)	CW(Q)
MaxCut	0.25					2857.8 (1)	5212.4 (10)		1121.0 (1)
	0.5					3246.2 (5)	3080.3 (6)		
	0.75					1436.1 (3)	3185.2 (9)		
GraphPart.	0.25	7179.0 (1)		5809.4 (3)	3457.0 (3)	3869.6 (1)	4060.5 (2)		3396.9 (1)
	0.5		7215.7 (2)	7987.9 (4)		6817.3 (1)	3569.4 (3)	3733.9 (1)	3853.2 (2)
	0.75			7669.1 (6)			4286.7 (3)		3961.7 (2)
MaxClique	0.25	14979.0 (3)		11520.8 (1)	8226.6 (2)			8292.9 (3)	10728.6 (2)
	0.5	683.1 (2)	6061.0 (3)	2612.8 (2)	5571.9 (2)	111.7 (1)		559.4 (2)	1754.3 (2)
	0.75	9.9 (1)	323.7 (1)	4.1 (1)	13825.2 (1)			58.6 (1)	20.2 (1)

First, we note that for the Maximum Cut and Graph Partitioning problems, classical solvers are unable to find the optimal solution, meaning we have to resort to the TBS measure. For Maximum Cut, neither of the Optimized, Random, SR(Q), SR(C) methods could compute the best known solution for any graph. Solely the optimized anneal offset feature (AO on qubit and chain level), and the optimized chain weights (CW for linear weights), can solve some of the graphs and attain best TBS measures. Overall, AO(C), annealing offsets at the chain level, gives the best performance for Maximum Cut.

For the Graph Partitioning problem, the optimized spin reversal can solve most problems on average, but only at the expense of incurring large TBS times. Using optimized anneal offsets on the chain level, as well as optimized chain weights (for quadratic couplers) yields best TBS times, however again at the expense of only solving few graphs which can bias the results. Spin reversal at the qubit level (SR(Q)) seems to be performing the best for this type of problem.

For the Maximum Clique problem, we are able to solve problem instances classically to optimality using the function `networkx.algorithms.clique.find_cliques` in Python’s Networkx package, and thus report TTS times. We observe that all methods can only solve around two of the 10 test graphs, and that overall, SR(Q) and CW(Q) yield the best TTS times for the Maximum Clique problem.

Apart from reporting TBS/TTS times, we can also evaluate all methods using the value of the Ising or QUBO formulation on the bitstring returned by D-Wave. Doing this for the best of the 30 embeddings on D-Wave with default

Table 2. Improvement (%) in the value of the QUBO/Ising formulation compared to Default-OE for the test graphs.

Problem	Density	Default-RE	SR(Q)	SR(C)	AO(Q)	AO(C)	CW(L)	CW(Q)
MaxCut	0.25	-0.3	-1.0	-0.7	4.2	8.8	0.8	-1.0
	0.50	0.2	-0.0	0.1	8.8	8.3	7.2	1.3
	0.75	0.0	-0.3	-0.7	9.0	9.9	4.4	0.1
GraphPart.	0.25	0.3	1.6	1.3	0.6	1.4	0.2	0.6
	0.50	-0.2	0.6	-0.6	0.0	0.4	0.1	0.5
	0.75	-0.2	0.7	-0.7	-0.4	0.6	0.4	0.2
MaxClique	0.25	-8.8	0.5	2.3	-4.6	6.4	-1.8	-6.6
	0.50	-0.6	1.2	2.3	-3.3	5.5	0.3	0.3
	0.75	-2.2	3.1	1.8	-1.4	0.3	0.1	0.2

parameters (i.e., Default-OE) allows us to set a reference point, and we report the improvement (in percent) of the obtained value over this reference in Table 2.

For the Maximum Cut problem, optimized anneal offsets (both on the qubit and the chain level) resulted in the largest percent improvement. For graph partitioning, spin reversal on the qubit level performs best, though the improvements are only of the order of one percent. Surprisingly, using Default-RE for Maximum Cut and Graph Partitioning does not perform very different than Default-OE. Lastly, for Maximum Clique, using anneal offsets on the chain level performs considerably better than the other techniques for density 0.25 and 0.5, with spin reversal being best for high densities.

4 Discussion

This work considered optimizing three recent features of the D-Wave 2000Q annealer, precisely spin reversal (on qubit or chain level), anneal offsets (on qubit or chain level), and chain weight distribution (for linear or quadratic couplers). After fixing the embedding, we perform a classical optimization over a suite of random test graphs using a differential evolution optimizer, and aim to investigate if it is possible to outperform the default D-Wave anneal setting with optimized parameters for the three techniques. Our overall aim is to tune SR, AO and CW such that these features work better on a whole class of problems.

We conclude that for random graphs, and the three NP-hard problems we considered, tuning anneal offsets indeed works best, yielding substantial improvements over the default D-Wave behavior, especially for the Maximum Cut and Graph Partitioning problems. Optimizing spin reversal and chain weights seems more dependent on the problem and measure.

This work leaves scope for a variety of future research avenues. First, we performed the optimization for SR, AO and CW individually, since each involves tuning around 2000 variables, and we found larger optimization problems to be infeasible. More elaborate optimization methods could allow us to optimize all

parameters simultaneously, potentially improving results. Second, it would be interesting to extend the experiments to more classes of NP-hard problems, with the aim to see how much the trained parameters of SR, AO, or CW differ, and to investigate if the trained SR, AO, or CW can be recycled for certain classes. Third, the fixed embedding setup could allow us to determine if certain hardware qubits behave more favorably if consistently spin reversed, or if consistently employed with a particular anneal offset. Finally, though time consuming, our experiments would benefit from larger sets of testing and training graphs.

Acknowledgments. This work has been supported by the US Department of Energy through the Los Alamos National Laboratory. Los Alamos National Laboratory is operated by Triad National Security, LLC, for the National Nuclear Security Administration of U.S. Department of Energy (Contract No. 89233218CNA000001) and by the Laboratory Directed Research and Development program of Los Alamos National Laboratory under project numbers 20190065DR and 20180267ER.

References

1. Andriyash, E., Bian, Z., Chudak, F.A., King, A.D., Macready, W.G.: Boosting integer factoring performance via quantum annealing offsets. Technical report, D-Wave Systems (2016)
2. Chapuis, G., Djidjev, H., Hahn, G., Rizk, G.: Finding maximum cliques on the D-wave quantum annealer. *J. Signal Process. Syst.* **91**(3–4), 363–377 (2019)
3. D-Wave Systems: Quantum Computing for the Real World Today (2000)
4. D-Wave Systems: Graph Partitioning QUBO (2019). https://github.com/dwave-examples/graph-partitioning/blob/master/graph_partitioning.py
5. D-Wave Systems: Maximum Cut QUBO (2019). https://github.com/dwave-examples/maximum-cut/blob/master/maximum_cut.py
6. D-Wave Systems: D-Wave System Documentation: Solving a Problem on the QPU - Using Spin-Reversal (Gauge) Transforms. Technical report, D-Wave Systems (2020)
7. D-Wave Systems: Technical Description of the D-Wave Quantum Processing Unit (2020)
8. King, A.D., Hoskinson, E., Lanting, T., Andriyash, E., Amin, M.H.: Degeneracy, degree, and heavy tails in quantum annealing. *Phys. Rev. A* **93**(5), 052320-1–052320-12 (2016)
9. Pelofske, E., Hahn, G., Djidjev, H.: Optimizing the spin reversal transform on the D-wave 2000Q. In: Proceedings of the 2019 IEEE International Conference on Rebooting Computing (ICRC), pp. 1–8 (2019)
10. Pelofske, E., Hahn, G., Djidjev, H.: Inferring the Dynamics of Ground-State Evolution of Quantum Annealers [arXiv:2009.06387](https://arxiv.org/abs/2009.06387), pp. 1–21 (2020)
11. Pelofske, E., Hahn, G., Djidjev, H.: Solving large maximum clique problems on a quantum annealer. In: Feld, S., Linnhoff-Popien, C. (eds.) QTOP 2019. LNCS, vol. 11413, pp. 123–135. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-14082-3_11
12. Pudenz, K.L.: Parameter setting for quantum annealers. In: 2016 IEEE High Performance Extreme Computing Conference (HPEC), pp. 1–6 (2016)
13. Storn, R., Price, K.: Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.* **11**, 341–359 (1997)
14. Virtanen, P., et al.: SciPy 1.0 contributors: SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nat. Meth.* **17**, 261–272 (2020)



A Behavioural Network Traffic Novelty Detection for the Internet of Things Infrastructures

Salma Abdalla Hamad¹(✉), Quan Z. Sheng¹, Dai Hoang Tran¹,
Wei Emma Zhang², and Surya Nepal³

¹ Department of Computing, Faculty of Science and Engineering,
Macquarie University, Sydney, Australia

{salma-abdalla-ibrahim-mah.h,michael.sheng,dai-hoang.tran}@mq.edu.au

² School of Computer Science, The University of Adelaide, Adelaide, Australia
wei.e.zhang@adelaide.edu.au

³ Data61, CSIRO, Sydney, Australia
surya.nepal@data61.csiro.au

Abstract. The Internet of Things (IoT) applied solutions are changing the way the world perceives technology. IoT devices are now being used in a wide range of applications to transfer or share relevant information, hence reducing human interventions. With such widespread IoT solutions, security becomes a significant concern. Many of the IoT devices are vulnerable due to several reasons, including in-secure implementations, poor life cycle management, and inappropriate configurations, leading to an increase in the risk of these devices getting exposed and attacked. However, the current security approaches for detecting compromised IoT devices are inefficient, especially for zero-day attacks. Since no one knows how a new attack would look like, it will be useful to monitor and detect anomalies using accurate detection techniques. This work probes the possibility of detecting IoT network traffic anomalies using novelty detection techniques; thus, it can detect compromised IoT devices. One of this work's main contributions is developing an IoT anomaly detection system named Behavioural Novelty Detection for IoT Infrastructure (BND-IoT). BND-IoT trains a neural network with novel selected behavioural features extracted from benign traffic only and then uses the novelty techniques to detect any unusual traffic patterns. We show that the presented approach effectively detects anomalies within IoT devices' network traffic with a robust average F1-score of 96.7% and a low false rejection rate of 7%.

Keywords: IoT security · Machine learning · IoT anomaly detection · Fingerprinting · Novelty detection · Outlier detection

1 Introduction

Billions of Internet of Things (IoT) devices are connected to each other, and the number is still increasing day by day. IoT has become a considerable element in

almost everybody’s daily life. With the expanded deployment of IoT solutions in almost every domain, security becomes a critical issue. Most of the IoT devices and solutions are vulnerable to attacks as they are neither updated frequently nor configured securely, e.g., using simple passwords or applying default configurations. It was reported recently [1] that attackers exploited insecure IoT devices and used these devices as bots to launch volumetric distributed denial-of-service (DDOS) and Brute force attacks.

Controlling IoT network traffic involves monitoring huge amount of moving data. Hence, it is easy for malicious activities to hide within the normal traffic and infiltrate the system without being detected, especially without the prior knowledge of such traffic patterns [2]. The main goal of a security solution is to have a scalable and reliable IoT infrastructure that is capable of detecting compromised IoT devices and malicious traffic within the network. Here, machine learning techniques can be used to detect malicious traffic traces and protect the IoT infrastructure [3]. In the recent years, machine learning techniques have proven their efficiency in many mission-critical applications, including anomaly and intrusion detection systems (IDS) [4]. A number of current security solutions use learning-based approaches, in which models are trained using comprehensive big datasets [4]. These trained models can be integrated with firewalls to improve the overall network detection and prevention solutions effectively.

Our work focuses on providing a solution that can detect malicious behaviours in IoT network traffic using anomaly detection techniques. In this paper, we present a solution named Behavioral Novelty Detection for IoT Infrastructure (BND-IoT). The goal of the BND-IoT system is to detect compromised IoT devices and malicious traffic within an infrastructure in real-time using novelty detection algorithms. The inputs of our detection model is a novel behavioural-based fingerprint, generated from normal traffic patterns only for each IoT device type. The detailed discussion of the novel fingerprinting technique and the novelty anomaly detection technique are presented in Sect. 4.

The key contributions of this research are as follows:

- We propose a novel behavioural fingerprinting technique that captures the behaviour patterns of Internet Protocol (IP)-enabled IoT devices network traffic. This behaviour-based fingerprint can be used by the novelty detection solution to catch malicious traffic, thus allowing the system to identify compromised IoT or rogue devices connected to the network.
- In the absence of any prior expert knowledge on anomalous data, we propose BND-IoT. BND-IoT is a real-time IoT network traffic anomaly detection system that can detect anomalous traffic from unknown, unseen attacks, and malware traffic, when the network model is trained with the normal traffic only. The aim is not only to detect known attacked patterns but also zero-day attacks with high detection rates (DRs) and low false positive rates (FPRs).

The rest of the paper is organized as follows. Section 2 overviews the related work on network anomaly detection. In Sect. 3, we present the threat model. We then propose BND-IoT which is an IoT device anomaly detection system

in Sect. 4. This section includes the details on the BND-IoT system architecture design and components. Section 5 reports the experimental studies and the results. Section 6 discusses the proposed solution and observed challenges. Finally, Sect. 7 concludes the work with possible future improvements.

2 Related Work

There is a significant amount of network traffic anomaly detection researches in the literature using machine learning techniques. Several approaches have been proposed to detect and prevent intrusions or anomalies in IoT infrastructures. A number of these proposals used two-class classification (benign and malicious) such as in [5,6]. This approach opposes the objective of an anomaly-based approaches that should detect any variation from the benign traffic behavior [7].

There are a number of fingerprinting approaches that extracts set of features that represent the behavior of IoT device types proposed in the literature, such as physical, wireless and network traces fingerprinting [8–10]. Fingerprinting IoT devices is usually a challenging task due to the high number of available device types and used protocols. Most of the existing network traces fingerprinting approaches that are used to feed into anomaly detection solutions [9,10] mainly focus on flow-based features or only include information from a limited number of network layers and protocols. Due to such limitations and the heterogeneous nature of IoT devices, these approaches are not applicable in real-time systems.

Lately, some deep learning (DL) approaches, especially recurrent neural network (RNN) were proposed that were trained with both normal and anomalous traffic with a similar set of features to detect anomalies, intrusion and malicious infected devices within a network, such as in [11,12]. In contrast to these researches, our work proposes an approach that models the normal traffic behaviors of IoT devices and uses novelty detection algorithms that train the network with normal benign traffic only. In general IoT devices have systematic behaviours, which usually connect to particular servers or do specialized tasks [13]. Accordingly, the novelty detection techniques can be useful, thus, behavioural deviations can be identified, disregarding the infection types. Moreover, the classifier can be trained faster and can detect unseen (zero-day attacks) anomalies in live network traffic.

In the literature, many one-class classification techniques have been proposed to identify network traffic intrusions or anomalies. A number of these techniques utilized the Support Vector Machines (SVM) methodology such as [2]. In [13], the authors tested four one-class classifiers to detect anomalies in IoT networks. The best achieved F1-score was 94%. The authors of [14] proposed a one-class classification approach based on RNN and federated self-learning technique. The classifier was trained with a sequence of symbols that represent normal benign traffic only that were collected at several security gateways. Each gateway was dedicated to monitor a client IoT network. On the one hand, the classifier in [14] was tested with anomalies generated from only Mirai malware [15]. On the other hand, this work used a dataset with generic type of attacks that can target any IoT device.

3 The Threat Model

We consider an adversary model, where malicious attackers are trying to compromise vulnerable IoT devices in the infrastructure. Attackers that compromise IoT devices can gain access to sensitive information on the compromised devices or use these devices as pivots or bots for further attacks. We also consider IoT devices in the infrastructure that are already compromised or infected with malware and produce unusual network behaviours.

In this paper we assume that the IoT devices are not compromised at the time of collecting the normal traffic. Accordingly, the normal network traffic of IoT devices collected for training the network is benign and free from any malicious traffic.

4 BND-IoT System

We propose BND-IoT, an IoT network traffic anomaly detection system to identify malicious traffic in the IoT infrastructure as well as compromised IoT devices within the network. To manage the challenge of high false positives (FP)s in IoT anomaly detection due to the heterogeneous nature of IoT, we propose a trained classifier for each IoT device type. Generating a behavioural baseline for each IoT device type’s normal network traffic, we then continuously monitor the network traffic for each of the connected IoT devices, to detect any anomalous traffic that deviate from the generated baseline. This detected anomalous traffic could be malicious traffic trying to compromise an IoT device or could be generated from a compromised IoT device within the infrastructure.

4.1 BND-IoT Architecture

The main components of the BND-IoT system architecture design include a fingerprinting solution to select and extract features and a classifier network, as shown in Fig. 1.

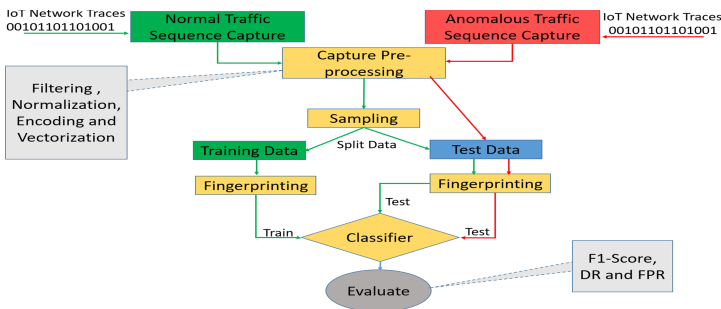


Fig. 1. The system model for IoT network traffic anomaly detection.

The network traces of IoT devices already connected to the infrastructure are monitored. The first component of the model is a packet capture that uses a random moving window to capture a sequence of packets from the monitored IoT devices network traces. This picked-up traffic is then applied to a pre-processing module to prepare the data before applying it to the fingerprinting solution. Pre-processing of data ensures all missing data (Null) are handled and the categorical data are converted to numeric and normalized. Then the pre-processed data is directed to the fingerprinting solution to select and extract features. The generated fingerprints are then fed into a classification network. This classification network is trained only with benign traffic. The classifier tries to match this traffic with the stored baseline for the IoT device traffic behaviors. The decision values of the classifier can be equal to 1 for normal benign traffic or either 0 or a negative value for unusual traffic traces. Since the classification is performed on a window of sequence of packets, the classifier triggers the detection of an anomaly only if the selected sequence contains a considerable number of anomalous packets, thus reducing the FPs. Unusual traffic traces or anomalous traffic indicate that the device may be compromised. Hence, this device should be isolated from the network and monitored for further analysis. This further monitoring and analysis can be performed by security event and information management (SEIM) solutions.

4.2 Feature Extraction

Based on the proposed feature selection technique in [16], we divide the data into windows of sequences of 30 packets for each IoT device. In [16] a fingerprint for each device was created by extracting flow-based features that include header and payload information from a number of network layers and protocols, such as Ethernet, IP and Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) information. Such information was captured from a sequence of IoT network packets. A 67-dimensional feature vector was used to represent unique network traffic features for each device type. The selected features include summary statistics such as minimum, maximum, mean and Fast Fourier Transforms (FFT) for a number of the selected features. In this work, we capture bi-directional device-device communication packets, while using the Media Access Control (MAC) address to identify the source and destination of packets. We generate a fingerprint for each device type using both behavioural and flow-based features.

On the one hand, to capture the behaviour of each IoT device type, we include information such as ports used and the usage frequency of these ports [5]. On the other hand, to capture information flow related data, we include information such as IP destinations and a number of traffic statistical information, such as Inter-arrival-time (IAT) [5]. To measure variability in data, we use statistical quartiles. Quartiles (q) are values that divide an ordered data into quarters [17]. Specifically, q1 is the calculation of the middle of the lower half of the data, q2 is the median and q3 is the calculation of the middle of the upper half of the data [17].

We choose to add a number of other selected features to uniquely present continuous network traffic for each IoT device type. We perform a number of experiments to choose the number of packets in the captured sequence to ensure enough information is captured without affecting the speed of fingerprinting and classifying the device traffic. Accordingly, we choose to capture a sequence of 30 packets from the network traffic of each IoT device. The proposed feature vector contains 74 features. All the selected features are described in Table 1.

Table 1. Features that are extracted to create a fingerprint for the normal traffic of each IoT device.

Number	Feature
0–17	Statistics (min, max, q1, median, mean, q3, var and iqr) of Packet Inter Arrival Times (IAT) and Fast Fourier Transforms (FFT) of IAT. IAT summary statistics and IAT 10 highest magnitudes of FFT. IAT is the measure of the delay between following packets
18–32	Statistics (min, max, q1, median, mean, q3, var and iqr) of Ethernet Frame and Packet Header Size. Statistics for the Ethernet frame and packet header size of transmitted packets in both-directions. Variations in sizes can detect volumetric attacks
33–48	Statistics (mean, min and max) of TCP Frequency, UDP Frequency, UDP Frequency, DHCP Frequency and DNS Frequency
49	Packet Rate. It is used to count the number of transmitted packets in a one minute period which is useful in detecting flooding attacks
50–55	Statistics (mean, min and max) of Packet Order and Payload Length. A count of the number of transmitted packets in each direction in a one minute period. Such info can help in identify amplification attacks
56–58	Periodic IP Destinations, Source and Destination Ports. A count of the number of IP destinations, source and destination ports within the selected window. These features can indicate an anomalous connection especially in distributed denial of service (DDOS)-attacks
59–61	Statistics (mean, min and max) of Packet Time To Live (TTL). TTL for TCP and UDP packets. TTL is the allowed hops count for each packet. In other words, this feature presents the lifespan of packets in a network
62–67	Statistics (mean, min and max) of TCP Window Size and TCP Payload Size. TCP window size represents the number of packets that are sent and acknowledged in a single acknowledgment by the receiver. This feature considers the device memory and processing speed
68–70	Statistics (mean, min and max) of TCP Dataofs. Summary statistics of TCP payload data offset. Data offsets communicate the data start point to the upper network layers. The authors of [16] presented the significant effect of this feature for identifying IoT device
71–73	Statistics (mean, min and max) of Packet Flags. Summary statistics of TCP flags. The frequency and statistics of occurrence of each flag can indicate the presence of a number of attacks, such as probing attacks

4.3 Classification Network

We formalize the problem of anomaly detection classification as a *one-class classification* problem [18], as we are training the network with benign traffic and expecting the network to classify network traffic as one for benign traffic and zero or negative for anomalous traffic depending on the type of the classifier.

There are different approaches including deep learning techniques that can be used as a base for performing one-class anomaly detection classification, specifically novelty detection for IoT network traffic. There are numerous traditional novelty detection techniques in the literature, which can be categorized into a number of categories, such as, distance-based and density-based [19]. However, these traditional techniques usually can not handle big data and data with different density regions [20]. An ideal anomaly detection classifier should i) be easy to tune, ii) have less parameters to set and have fast run-time, iii) can scale up to handle big data, and iv) be simple and consistent with different types of data and different types of anomalies [20]. Isolation Forest (IF) [21] is one of the novelty detection approaches that can handle big data challenges, and can be easily tuned and used with different types of data and anomalies. IF focuses on separating anomalous data, not on trained normal traffic. It usually consists of an ensemble of trees, where partial models and sub-sampling are used to isolate anomalies, with low computations and linear time in comparison to traditional distance and density-based novelty detection approaches [21]. The ability of using sub-sampling enhances the possibility of hosting a fast execution online anomaly detection solution that uses low memory to fit big data infrastructures [21]. IF separates anomaly instances which are usually closer to the root of the tree while the normal traffic instances are more likely to be at the deeper ends [22]. Anomaly score $S(x,n)$ for each instance reflects the possibility of being an anomaly and is calculated using the average path length from the IF trees as described in Eq. (1) [21]:

$$S = \frac{2 \cdot E(h(x))}{c(n)}, \quad E(h(x)) = \frac{1}{L} \cdot \sum_{i=1}^L h_i(x) \quad (1)$$

where x is the test sample, L indicates the number of trees in the forest, $E(h(x))$ is the calculation of the average $h(x)$ for the ensemble of trees, n is the number of subsampling size, $c(n)$ is the calculation of the average $h(x)$ for a given n and h_i represents the length of the i^{th} tree.

In this paper, we are interested to use Isolation Forest (IF) as the novelty classifier and we examine and compare the results of three other approaches for novelty detection, which are, Local Outlier Factor (LOF), Elliptic Envelope (EE), and RNN architectures. We set novelty to true for IF, LOF and EE by setting the contamination to zero for IF and EE classifiers and setting the novelty parameter to true for the LOF classifier. In the RNN classifier we use a fully-connected layer followed by a sigmoid regression layer. The dimension of the fully-connected layer is set equal to the feature dimension. The sigmoid layer has two outputs to represent the normal traffic and the abnormal traffic. All classifiers are trained with normal traffic to ensure classifiers learn benign traffic

only. We expect that Long Short-Term Memory (LSTM) can also provide better results than RNNs. This is due to the fact that LSTMs control the vanishing gradient issue faced by RNN [23]. However, defining a time step as well as a base to identify unusual behaviour is an issue for this technique.

5 Implementations and Results

This section presents the analysis and evaluation of the BND-IoT system.

5.1 DataSet

In our experimental studies, we choose to use the UNSW IoT analytics dataset [12]. To the best of our knowledge, it is one of the latest and the most comprehensive IoT traffic datasets publicly available at the time of this writing. The dataset includes collected and synthesised network traffic traces for 30 different IoT devices connected to the UNSW lab infrastructure [12]. Normal traffic flow is collected in 26 different packet capture files (PCAP files). While running the attacks experiments, the authors of [12] collect 17 different PCAP files that capture both attack and normal traffic traces. We use the subset that is released for the community use. This community use subset contains normal and attack traffics of nine different IoT devices. We refer the readers to [12] and [24] for a detailed description of the dataset.

To use the UNSW IoT analytics dataset, we first filter the traffic for each device separately using both device MAC address and IP address. These filtered PCAP files for each device are then separated into normal traffic and attack traffic traces. The PCAP files that have the network traffic traces are huge in size, accordingly pre-processing is needed (e.g., read, split files per session, normalize) to enhance the massive amount of network traffic. We select features from the normal traffic traces to reduce the size after vectorizing the dataset to 74 features by using behavioral fingerprint for each device. Those features represent unique network traffic behaviour fingerprint for each device. The extracted features are then processed and serialized to be saved into files to enhance the management of the data. The processed data is divided into training and validation sets to be used for learning and prediction of normal behaviors, respectively. The same procedure is applied to attack traces and the extracted behaviour fingerprints are saved in separate files to be used for testing and evaluation.

5.2 Experimental Setup

We use the network traffic from nine different IoT devices. We train the neural networks with normal traffic filtered to detect traffic of each device type separately. We develop our attack detection solution using Python programming

language libraries (e.g., Pandas, Numpy, sklearn, keras). We divide the benign traffic using the train-test-split module from scikit-learn library [25], into 80% for training and 20% for validation. The attacked traffic dataset is used for testing the accuracy of the proposed solution, while the validation normal traffic is used to identify the false positive rate (FPR).

Evaluation Metrics. This work is presented as a one class classification problem based on binary classification to classify data as normal or attack. DR (detection rate) is a representation of correctly identified anomalies or can also be called True Positives (TP), True Positive Rate (TPR), sensitivity or recall. To detect the percentage of the incorrect classification of normal traffic as malicious, we utilize FPR. Classification report and receiver operating characteristics (ROC) are used to show the accuracy (TP and True Negative (TN)) and correctness of the proposed model.

5.3 Preliminary Results

We conduct the performance evaluation of our anomaly detection solution first considering IoT devices individually and then all together. Figure 2 illustrates the experimental results done.

The performance results (F1-score) for different novelty detection techniques are shown in Fig. 2-(a). The results show that IF classifier presents the best results for most of the devices in comparison to the other three approaches.

We are able to detect 99.9% of all anomalous or malicious traffic using IF, in other words, achieving true positive rate (TPR) of 99.9%. Figure 2-(b) demonstrates the ROC curve of FPR and TPR for all IoT devices. The figure shows 95% or higher TPR while maintaining a low FPR, which is one of the goals of our work. Figure 2-(c) demonstrates the IF classification report for all IoT devices. Classification report shows the precision, recall and F1-score for each IoT device. The overall F1-score achieved is 96.7%.

We use the validation data for calculating FPR. Any trigger for anomaly in this data will indicate a FP as the validating data contained only benign traffics. Figure 2-(d) demonstrates the DR versus False Rejection Rates (FRR) for all devices. The malicious traffic DR for most of the devices is almost perfect except for the Wemo motion sensor that have anomaly detection of 99%. Regarding the FP, we achieve 2–7% FPR. Knowing that in real IoT network settings that include huge number of connected devices, this FPR may not be very efficient. The higher the FPs, the more false alerts generated for the infrastructure administrators to investigate.

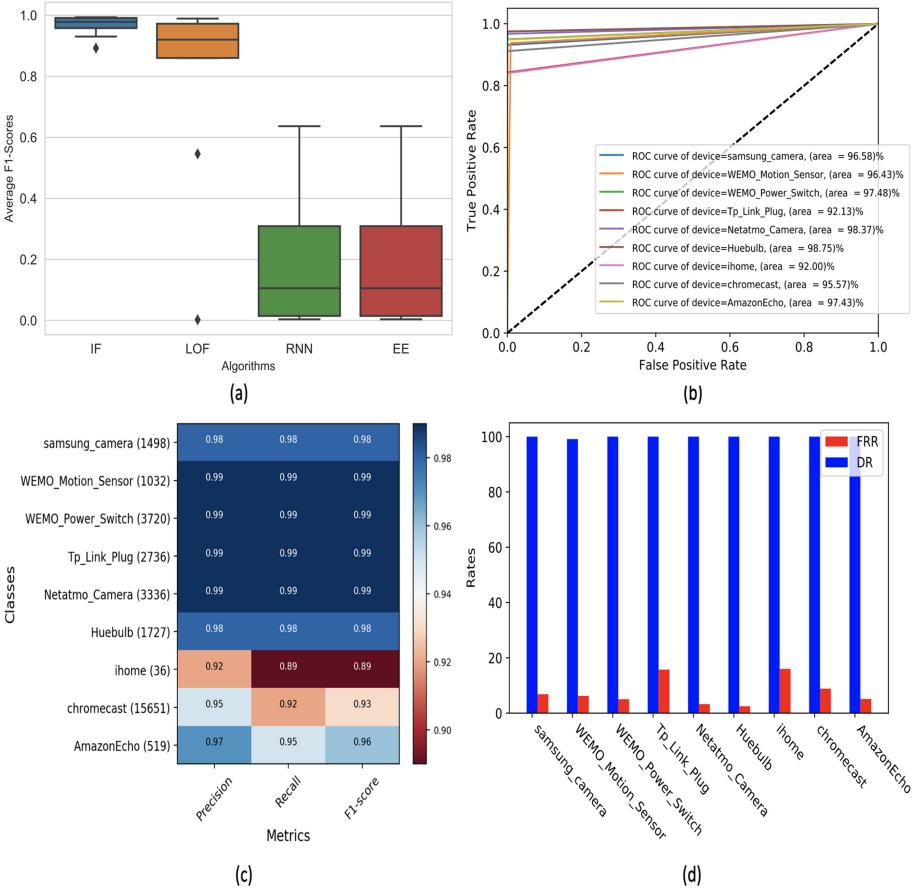


Fig. 2. (a) Comparison between the Four Novelty Techniques when Tested with the Nine IoT Devices. IF is the best achiever with average F1-score of 96.7%. (b) ROC Curve for the IoT Devices using IF Classifier. ROC curve shows the TPR and FPR for each of the IoT devices. The area under each of the curves is also calculated and illustrated. (c) Classification Report for the IoT Devices using IF Classifier. The highest F1-score is 99% for Wemo motion sensor, Wemo power switch, Tp link plug and Natetmo camera classifiers. While the lowest F1-score is 89% for Ihome device classifier. (d) Anomalous Traffic DRs Vs (FRR) for All Devices. The blue bars represent the malicious DR, showing almost perfect detection performance. The red bars represent the FPR. (Color figure online)

6 Discussions

One of the early challenges that we face in this work is finding a suitable dataset that perfectly represents an actual IoT network traffic with abundant training and testing data. Most of the available datasets use either synthetic IoT data or do not include IoT network traffic. Accordingly, we decide to use the IoT dataset from [24] to prove our concept.

Observations. In our experiments the FPR for most of the IoT device types is low, except for the TP-Link plug and the IHome devices. The difference in the FPR for these two devices encourage us to analyze the dataset and investigate the reasons for the higher FPRs. The Huebulb and the Natatmo camera are observed to have the lowest FPR. By analyzing the training data used for these devices, we find out that these two devices has more training data instances in comparison with the training data instances used for the devices that produce high FPRs. The benign instances that we use for training and testing the FPs on the four device are as below:

- 634 and 147 instances for training and testing the FPs of the Tp link plug and Ihome classifiers respectively.
- 3,966 and 9,171 instances for training and testing the FPs of the Natatmo camera and Huebulb device classifiers respectively.

We expect that including more training data for these devices can reduce their FPRs.

Scalability. This work dedicates a trained classifier for each type of IoT devices, to ensure that the proposed solution is scalable and independent on the number of IoT device types connected to the industrial infrastructure. Adding any new type of IoT devices to the current infrastructure, will not affect the performance of the other classifiers within the infrastructure.

Generalization. The generated fingerprints are based on behaviour information independent on the communication protocols and the dataset used, which makes the fingerprinting technique useful for different datasets and environments.

Comparison with Other Techniques. Our work considers more details of the IoT fingerprinting process and steps used as well as pre-possessing of data in comparison to other works. Moreover, our work relies only on normal traffic traces to train the classifier, making it possible to identify any unseen malicious traffics. There are other works in the literature that target one-class approach to identify anomalies. Ideally, a logical comparison should happen between models that use the same dataset and/or same evaluation metrics. The state-of-the-art approaches use unpublished proprietary IoT dataset. However, we discuss the differences between the state of the art approaches and our approach, by using a publicly available IoT dataset. On the one hand, the authors of [14] extracted feature from each network packets and mapped them into symbols from a sequence of 250 packets to create a fingerprint for each device, and used Gated Recurrent Units (GRUs) [26] for identifying anomalies. The dataset included malicious traffic generated from Mirai malware [15]. They achieved a 94.1–95.6% DR and 0–1% FPR. On the other hand, we exploit 74 features from only a sequence of 30 packets to create a unique behavioural fingerprint. The fingerprint is then applied to a novelty detector, achieving an average 99.9% TPR and average 7% FPR.

7 Conclusion and Future Work

This paper presents a solution for a conceptual framework to encourage discussions among the security community to direct more research efforts for enhancing the security of the emerging IoT infrastructures. We propose the BND-IoT system to detect malicious traffic within an IoT infrastructure. Our proposed BND-IoT system is based on a novel behavioural feature selection scheme along with the novelty detection techniques. More specifically, we exploit machine learning and deep learning algorithms as base for detecting malicious or unusual IoT device communications using novelty detection techniques. The main objective from using the novelty detection techniques is to detect unseen malicious traffic, even without training the network for similar anomalous traffic. The results demonstrate that our BND-IoT system can achieve 99.9% anomalous traffic detection and an average of 7% false rejection rate.

In the future, we will continue to improve the proposed solution on false positives. We also plan to further develop our model with different classification algorithms such as Conventional Neural Network (CNN) and LSTMs and compare the results. Another promising direction that we present in this work and shows promising results with very high DRs and low FPs, is the use of RNNs as a novelty classifier, given its capability to take into consideration on recent historical data. Moreover, we will work on increasing the training data for the classifier and include traffic from different IoT device types to ensure that the solution is suitable for end-to-end IoT infrastructure anomaly detection.

References

1. Boddy, S., Shattuck, J.: The hunt for IoT: the rise of the thingbots. *F5 Labs Threat Anal. Rep.* **3**, 1–27 (2017)
2. Lamrini, B., Gjini, A., Daudin, S.: Anomaly detection using similarity-based one-class SVM for network traffic characterization. In: *International Workshop on Principles of Diagnosis - DX*, pp. 1–8 (2018)
3. Hasan, M., Islam, M.M., Zarif, M.I.I., Hashem, M.: Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. *IEEE Internet Things - IoT J.* **7**, 1–14 (2019)
4. Salman, T., Bhamare, D.: Machine learning for anomaly detection and categorization in multi-cloud environments. In: *IEEE International Conference on Cyber Security and Cloud Computing - CSCloud*, pp. 97–103 (2017)
5. Moustafa, N., Turnbull, B., Choo, K.K.R.: An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of Internet of Things. *IEEE Internet Things J. - IoT J.* **6**, 4815–4830 (2018)
6. Feng, F., Liu, X., Yong, B., Zhou, R., Zhou, Q.: Anomaly detection in ad-hoc networks based on deep learning model: a plug and play device. *Ad Hoc Netw.* **84**, 82–89 (2019)
7. Deng, J., Dong, J.: Imagenet: a large-scale hierarchical image database. In: *IEEE Conference on Computer Vision and Pattern Recognition - CVPR*, pp. 248–255 (2009)

8. Franklin, J., Mccoy, D., Tabriz, P., Neagoie, V., Randwyk, J.V., Sicker, D.: Passive data link layer 802.11 wireless device driver fingerprinting. In: USENIX Security Symposium - USENIX-SS, p. Article No. 12 (2006)
9. Radhakrishnan, S.V., Uluagac, A.S., Beyah, R.: GTID: a technique for physical device and device type fingerprinting. *IEEE Trans. Dependable Secure Comput.* **TDSC 12**, 519–532 (2015)
10. Miettinen, M., Marchal, S., Hafeez, I., Asokan, N., Sadeghi, A.R., Tarkoma, S.: IoT SENTINEL: automated device-type identification for security enforcement in IoT. In: IEEE International Conference on Distributed Computing Systems - ICDCS, pp. 2177–2184 (2017)
11. Mirsky, Y., Doitshman, T., Elovici, Y.: Kitsune: an ensemble of autoencoders for online network intrusion detection. In: Network and Distributed Systems Security Symposium - NDSS, pp. 1–15 (2018)
12. Hamza, A., Gharakheili, H.H., Benson: Detecting volumetric attacks on IoT devices via SDN-based monitoring of MUD activity. In: ACM Symposium on SDN Research - SOSR, pp. 36–48 (2019)
13. Bezerra, V.H., da Costa, V.G.T., Barbon Junior, S., Miani, R.S., Zarpelão, B.B.: IoTDS: a one-class classification approach to detect botnets in Internet of Things devices. *Sensors* **19**, 1–20 (2019)
14. Nguyen, T.D., Marchal, S., Miettinen, M., Fereidooni, H., Asokan, N.: DIoT: a federated self-learning anomaly detection system for IoT. In: IEEE International Conference on Distributed Computing Systems - ICDCS, pp. 1–12 (2019)
15. Fruhlinger, J.: The Mirai botnet explained: how IoT devices almost brought down the internet—CSO Online (2018). <https://www.csoonline.com/article/3258748/the-mirai-botnet-explained-how-teen-scammers-and-cctv-cameras-almost-brought-down-the-internet.html>
16. Hamad, S.A., Zhang, W.E., Sheng, Q.Z., Nepal, S.: IoT device identification via network-flow based fingerprinting and learning. *IEEE International Conference on Trust, Security and Privacy in Computing and Communications - TrustCom*, pp. 103–111 (2019)
17. Pierce, R.: Quartiles (2016). <https://mathsisfun.com/data/quartiles.html>
18. Bellinger, C., Sharma, S., Japkowicz, N.: One-class versus binary classification: which and when? In: International Conference on Machine Learning and Applications - ICMLA, pp. 102–106 (2012)
19. Pimentel, M.A.F., Clifton, D.A., Clifton, L.: A review of novelty detection. *Signal Process.* **99**, 215–249 (2014)
20. Ting, K.M., Aryal, S.: Tutorial: which anomaly detector should i use? In: IEEE International Conference on Data Mining - ICDM, pp. 1–143 (2018)
21. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: IEEE International Conference on Data Mining - ICDM, pp. 413–422 (2008)
22. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data TKDD* **6**, 1–37 (2012)
23. Guru99: RNN(Recurrent Neural Network) Tutorial: TensorFlow Example (2020). <https://www.guru99.com/rnn-tutorial.html>
24. UNSW-Sydney: IoT Security - IoT Traffic Analysis (2019). <https://iotanalytics.unsw.edu.au/attack-data>
25. Scikit: Sklearn documentation (2019). <https://scikit-learn.org/stable/>
26. Keras: Home - Keras Documentation (2019). <https://keras.io/>



A Fast Algorithm for Image Segmentation Based on Global Cosine Fitting Energy Model

Yanping Chen¹, Le Zou¹, Zhize Wu², Qianjing Huang³,
and Xiaofeng Wang¹✉

¹ School of Artificial Intelligence and Big Data,
Hefei University, Hefei 230601, Anhui, China
Xiao-FengWangxfwang@hfu.edu.cn

² Institute of Applied Optimization, School of Artificial Intelligence
and Big Data, Hefei University, Hefei 230601, Anhui, China

³ College of Bioengineering Food and Environmental Sciences,
Hefei University, Hefei 23060, Anhui, China

Abstract. Image segmentation plays an important role in computer vision and image processing. Level set model is a classical image segmentation method. During level set evolution, almost all the level set energy minimization of image segmentation are based on the gradient descent method and the finite difference scheme. The speed of evolution is slow and easy to fall into local minima. In this paper, we propose a fast sweeping optimization algorithm to minimize global cosine fitting (GCF) model. When moving a pixel from the one side region to the another side region of evolving contour, the sweeping algorithm calculates the energy change directly and checks whether the cosine fitting energy is decreased. With this, we can avoid solving the Euler-Lagrange equation and the partial differential equation, which usually take a lot of time. Moreover, our proposal is robust to initial level set contour and it automatically handles the topological variety, algorithm automatic termination and no longer requires the reinitialization step, parameter adjustment and the distance regularization term. The experiments on real noise and synthetic images show the effectiveness of the sweeping algorithm.

Keywords: Image segmentation · Global cosine fitting model · Level set · Sweeping algorithm

1 Introduction

Computer vision is an interdisciplinary scientific field of artificial intelligence (AI), which deals with how computers can be made to gain high-level understanding from digital images or videos. Image segmentation is a fundamental and key problem in image processing and computer vision [1, 2]. The goal of segmentation is to divide an image into regions which is much easier and meaningful to analyze. During the past decades, thousands of traditional image segmentation methods have attracted scholars' attentions.

The active contour models is a classical traditional image segmentation method, the level set methods (LSMs) is one famous kind of active contour models. The LSMs can be roughly divided into edge-based models [2], region-based models [3] and hybrid

models [4]. Edge-based models usually rely on the gradient features of the image to construct their main external driving forces. Distance regularized level set evolution (DRLSE) is the famous one of them proposed by Li et al. [5]. The region-based models have a better result on segmenting images with weak boundaries and are less sensitive to initial conditions mostly when compared with edge-based methods. The Chan-Vese (CV) model [3] is the most famous one, it is particularly efficient for images containing homogeneous regions with distinct intensity means. However, CV model can't segment the image with intensity inhomogeneity. The CV model is sensitive to the placement of initial contour and setting of initial parameters. To improve the performance of global region-based methods, some local region-based methods and some new region based level set models were proposed. The classical ones include the region-scalable fitting (RSF) model [6], the local Chan-Vese (LCV) model [7], the local image fitting (LIF) model [8] and the local intensity clustering model (LIC) [9] etc. Min et al. [10] proposed a local salient fitting (LSF) model to effectively segment those images with severe intensity inhomogeneity. Wang et al. [11] presented a level set image segmentation by employing the cosine function to measure the data fitting term of the CV model (GCF).

The traditional numerical methods usually use the gradient descent (GD) and the finite difference to solve level set energy functional. The algorithm is simple, easy to understand and implement by computer programming. But the evolution equation of the level set function requires a large amount of computation, which limits the application of the level set method in practice. Meantime, such numerical implementation methods have some obvious disadvantages, i.e., time consumption is often too high to meet the high requirements of evolution speed in some special applications. In order to improve the speed of level set evolution equation, many researchers have shown an increased interest in numerical implementation methods of evolution equation. For example, narrow band method [12], the fast marching method [13], additive operator splitting (AOS) [14], difference scheme and so on. Many new and efficient methods have been proposed to solve the regional level set image segment model in recent years. Yang et al. [15] developed an improved level set method to accelerate the evolution of curves. Wang et al. [16] proposed the Hermite difference operator instead of the finite difference method, but the accuracy is still not high enough. Song et al. [17] presented a fast algorithm to solve the Chan Vese image segmentation model with less sweep. He et al. [18] generalized the above algorithm to solve the Chan Vese model by multiphase level set methods. Zou et al. [19] studied the pre-sweeping algorithm for solving the Chan-Vese model. The speed is very fast and keeps all the advantages of level set method. The sweeping algorithm can be easily extended to arbitrary finite dimensional image segmentation. Krinidis et al. [20] generalized the sweeping algorithm to the fuzzy energy-based active contour model, and get better image segmentation performance. Shyu et al. [21] proposed an energy functional including a local fuzzy energy and a global fuzzy energy to attract the active contour

and stop it on the object boundaries. Based on gaussian mixture model (GMM) intensity distribution estimator, Shyu et al. [22] constructed a fuzzy energy functional image segment method. Boutiche et al. [23, 24] and Zou et al. [25] presented the sweeping optimization algorithm to achieve fast segment results based on the level set image segment model. This method does not need to meet the stability conditions, and the parameter setting is simple. However, for strong noise images, severe intensity inhomogeneous images, complex scene image, it can not obtain a satisfactory effect. Zou et al. [26] studied a new Local Chan–Vese (LCV) model by using the cosine function to measure the data fitting term in traditional level set image segment models and present a new distance regularized based on polynomial function. Boutiche et al. [27] proposed a fast multi-channel implicit active contour method for performing soil and plant segmentation in color agriculture images.

Many new improved CV model and efficient methods have been proposed to solve the Chan-Vese model in recent years. In this paper, we described a new fast algorithm by calculating a variational energy based on the global cosine fitting energy functional. To minimize the energy functional, instead of solving the Euler–Lagrange equation, the cosine fitting energy are calculated directly by the fast sweeping optimization algorithm for level set energy functional optimization. The proposed algorithm therefore overcomes the initialization problem of the gradient descent (GD) based active contour model and significantly improves the computational speed. Furthermore, instead of computing the associated Euler–Lagrange equation, we apply a direct method to solve the corresponding partial differential equation without numerical stability constraints. The algorithm is very fast and robust to the initial placement of level set contour and setting of the parameters. Experiments are given to show the proposed algorithm have better performance than the traditional algorithms. Although the examples considered in this paper are 2-D gray images, it can extend the algorithm to higher dimensional image segmentation problems.

The remaining of this paper is structured as follows: in Sect. 2, the related model (GCF) is introduced. In Sect. 3, a fast algorithm based on sweeping principle algorithm for global cosine fitting energy functional is presented. In Sect. 4, some examples on real and synthetic images are given to demonstrate the robustness and the efficiency of the proposed algorithm. Finally, some conclusions are provided in Sect. 5.

2 Related Model

To get a better segment performance of the image with intensity inhomogeneity, Wang et al. [11] present a global cosine fitting energy by use of cosine fitting. Let $\Omega \subset \mathbb{R}^2$ be the two-dimensional image domain and $I : \Omega \rightarrow \mathbb{R}$ be the intensity image. For a given grayscale image, Wang et al. [11] change the given image to a gray scale image on the interval $[0, 1]$. For a given grayscale image on the interval $[0, 1]$, the cosine function

error preserves smaller error and more gentle than L2 norm on the interval $[0, 1]$, so we think the cosine fitting energy is less sensitive to the interference of the noise and robust to low contrast than the fitting energy in the CV model. Wang et al. [11] constructed the global cosine fitting energy as follows:

$$\begin{aligned}
 E(c_1, c_2, \phi) &= \lambda_1 \int_{\Omega} (-\cos(I - d_1))H(\phi(x))dx \\
 &+ \lambda_2 \int_{\Omega} (-\cos(I - d_2))(1 - H(\phi(x)))dx \\
 &+ \mu \cdot \int_{\Omega} \delta(\phi(x))|\nabla\phi(x)|dx + \nu \int_{\Omega} \frac{1}{2} (|\nabla\phi(x)|)^2 dx,
 \end{aligned} \tag{1}$$

where λ_1, λ_2 are fixed parameters, d_1 and d_2 are two constants that approximate the image inside and outside C , respectively. It is worth noting that d_1 and d_2 are on the interval $[0, 1]$, and the image I is also on the interval $[0,1]$ in the rest of this paper. The minimization of (1) can be solved by taking the Euler-Lagrange equations and updating $\phi(x)$ according to the gradient descent method as the Eq. (2):

$$\frac{\partial\phi}{\partial t} = -\delta(\phi)[\lambda_1 \cdot (-\cos(I - d_1)) - \lambda_2 \cdot (-\cos(I - d_2))] + \mu \cdot \delta(\phi) \operatorname{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \nu \nabla\phi, \tag{2}$$

At each iteration, d_1 and d_2 can be computed as follows:

$$d_1(\phi) = \arctan \frac{\int_{\Omega} \sin(I(x))H(\phi(x))dx}{\int_{\Omega} \cos(I(x))H(\phi(x))dx}, d_2(\phi) = \arctan \frac{\int_{\Omega} \sin(I(x))(1 - H(\phi(x)))dx}{\int_{\Omega} \cos(I(x))(1 - H(\phi(x)))dx} \tag{3}$$

Generally, the usual approach to find the solution of minimization energy functional problem as in (1) is to derive its Euler–Lagrange equation and then to use explicit finite difference to solve the above equation [11]. However, due to the numerical stability constraint, only the small time step can be selected. The numerical algorithm is often not efficient and the speed is slow, the parameter setting are very complex. This might lead to the fact that to get the precisely desired results, the time step needs to be small enough. Therefore the computational complexity is higher, and the convergence rate is lower. In the above evolution equation, it is necessary to calculate the differential and curvature, which brings great complexity and time. To overcome above problem, in this paper, we describe a fast and robust algorithm.

3 Fast Algorithm to Minimize the Global Cosine Fitting Energy Model

In the paper [11], the authors gave a finite difference scheme of the global cosine fitting energy model. However, the finite difference scheme in level set method has some drawbacks, such as larger approximation error and time-cost consuming and so on. To avoid the slow evolution and time-consuming computation in traditional level set image segmentation model, and inspired from the work published by Song[17] and Boutiche [23, 24], we propose a sweeping optimization principle algorithm to minimize the global cosine fitting energy functional (1). The proposed algorithm need not to solve any partial differential equations (PDE) and have not satisfy numerical stability conditions. We only sweep all the pixels of the given image, and then test each pixel to check whether the energy decreases or not when we change a pixel from the inside of the curve to outside and vice versa. In our algorithm, we needn't compute the time-consuming curve curvature.

Firstly, in order to be able to make use of the cosine similarity measure, as in the paper [11], the given gray scale image was changed into a gray scale image on the interval [0, 1]. In the global cosine fitting energy functional measure (1), the curve evolution is dominated by the fidelity data term which can be written as follows:

$$F = -\lambda_1 \cos(I(x) - d_1) + \lambda_2 \cos(I(x) - d_2) \quad (4)$$

Remind that the Eq. (4) indicates the global region fitting energy. However, the inside and outside energy was deducted as follows:

$$\begin{cases} -\cos(I(x) - d_1) & \text{if } \phi(x) > 0 \\ -\cos(I(x) - d_2) & \text{if } \phi(x) < 0 \end{cases} \quad (5)$$

Suppose the object is represented by A , the background is B , the corresponding value for A and B is a and b . Given an initial partition $\phi > 0$ and $\phi < 0$, denoted by ϕ_1 , ϕ_2 . Assume there are m points in ϕ_1 and n points in ϕ_2 . Let d_1 , d_2 , F_1 , F_2 be the average and total energy for ϕ_1 , ϕ_2 . Consider a given point $Q \in I$, for convenience of calculations, assume $Q \in \phi_1$ and the intensity value of point Q is x . If we change Q from ϕ_1 to ϕ_2 , let \hat{d}_1 , \hat{d}_2 be the new average for ϕ_1 and ϕ_2 respectively, and \hat{F}_1, \hat{F}_2 be the new energy for ϕ_1 and ϕ_2 . Then we can easily calculate:

$$\hat{d}_1 = d_1 + \frac{d_1 - x}{m - 1} \quad (6)$$

$$\hat{d}_2 = d_2 - \frac{d_2 - x}{n + 1} \quad (7)$$

From the Eq. (5), we can get the old and new energy according to the energy inside and outside of the evolution curve. We can construct a compute method of the cosine fitting energy change as follows:

$$\hat{F}_1 = F_1 - \lambda_1(-\cos(I(x) - d_1)) \frac{m}{m-1} \quad (8)$$

$$\hat{F}_2 = F_2 + \lambda_2(-\cos(I(x) - d_2)) \frac{n}{n+1} \quad (9)$$

If Q change from ϕ_2 to ϕ_1 , that is from inside of the curve to outside of the curve, we can also calculate the change of energy from outside of the curve to inside the curve. The difference between the new energy and old energy is:

$$\Delta F_{21} = \lambda_1 \left[(-\cos(I(x) - d_1)) \frac{m}{m+1} \right] - \lambda_2 \left[(-\cos(I(x) - d_2)) \frac{n}{n-1} \right] + vP \quad (10)$$

Similarly, if Q change from ϕ_1 to ϕ_2 , the change of total energy is:

$$\Delta F_{12} = -\lambda_1 \left[(-\cos(I(x) - d_1)) \frac{m}{m-1} \right] + \lambda_2 \left[(-\cos(I(x) - d_2)) \frac{n}{n+1} \right] + vP \quad (11)$$

where m and n are the area (number of pixels) inside and outside of the initial level set curve ϕ respectively, v is a nonnegative small constant. We add a length term P to increase the performance of segmentation with noisy images. For intensity image, we don't need the length term.

Similar to the method of the algorithm [23, 24], the length term P is approximated by Eq. (12).

$$P = \sum_{i,j} \sqrt{(H(\phi(i+1,j)) - H(\phi(i,j)))^2 + (H(\phi(i,j+1)) - H(\phi(i,j)))^2} \quad (12)$$

Different the traditional finite difference scheme, the time step is not restricted as in the present sweeping optimization algorithm. For each pixel in a certain iteration, the ϕ of (11) is used to calculate the change in the value of the energy functional. If the energy decreases, the ϕ will be change the sign, otherwise, the ϕ keeps the same.

The main steps of the proposed sweeping optimization principle algorithm for the global cosine fitting energy model are given as follows:

-
- Step 1: Change the given gray scale image into a gray scale image on the interval $[0, 1]$. Place any the initial level set on the given image. Initialize the level set function $\phi=1$ for one part and $\phi=-1$ for another one.
- Step 2: Initialize the parameters including the number of scales ε (for computing) and k, v ,
Set F is a matrix of size ϕ set to 1.
- Step 3: Compute d_1, d_2 by equations (6),
Compute m, n which are the area (number of pixels) inside and outside of the evolving curves for $\phi=1$ and $\phi=-1$ for each pixel of ϕ do
If $\phi=1$ then compute the difference between the new and old energy ΔF_{12} as shown in equation (14),
If $\Delta F_{12} < 0$ then change ϕ from +1 to -1.
If $\phi=-1$ then compute the difference between the new and old energy ΔF_{12} as shown in equation (13),
If $\Delta F_{21} < 0$ then change ϕ from -1 to +1.
 $F = \Delta F_{21} + \Delta F_{12}$,
- Step 4: Repeat the step 3 until the total energy F remains unchanged.
-

From the above steps we can see the proposed algorithm sweeps all the pixels of the image and then checks the energy variation for each pixel when the pixel is moved from the inside of the evolving curve to the outside and vice versa. The algorithm avoids solving any derivative calculations, partial differential equation and does not require any numerical stability conditions. Consequently, there is no need for F to be differentiable and the distance regularization term. Furthermore, sweeping principle algorithm allows using a binary level set function during the minimization process instead of the signed distance function, and avoids its negative effects and speeds up the optimization process. The algorithm has many advantages, for example, robust to initial level set contour, automatically handle the topological variety, algorithm automatic termination. There is no need to the reinitialization step, parameter adjustment, any stability conditions, so the segment speed dramatically improve.

4 Experiments

In this section, we present several examples to show the effectiveness and efficiency of the sweeping optimization principle algorithm for the global cosine fitting energy model. For simplicity, unless specifically mentioned, we set $\lambda_1 = \lambda_2 = 255$ and $\varepsilon = 1, v = 0$ in all the examples for the proposed sweeping algorithm. The proposed model was programmed using MATLAB 2017a and performed on an Intel Core (TM) i7-7700 3.6 GHz CPU, 8G RAM, and 64 bit Windows 10 operating system. In all the

experiments, all the images use the cycle as an initial contour, the solid red lines are used to represent the contours in each iteration.

In the first experiment, the perfect segmentation is obtained for three clean (without noise) synthetic images with slight intensity inhomogeneous as shown in Fig. 1. It is worth noting that the segmentation speed is very fast, it can obtain better segment performance in only 2 sweeps on both images, which demonstrates the effectiveness and efficiency of the proposed algorithm.

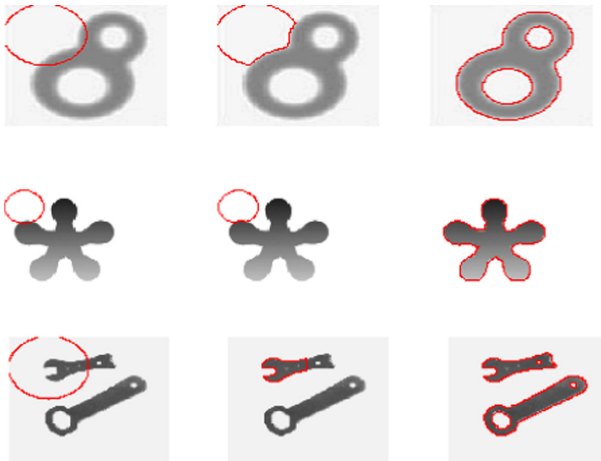


Fig. 1. Segmentation results on three clean (without noise) synthetic images with slight intensity inhomogeneous: first column, initial contours; second column, results of the first iteration of sweeping; third column, results of the second iteration of sweeping.

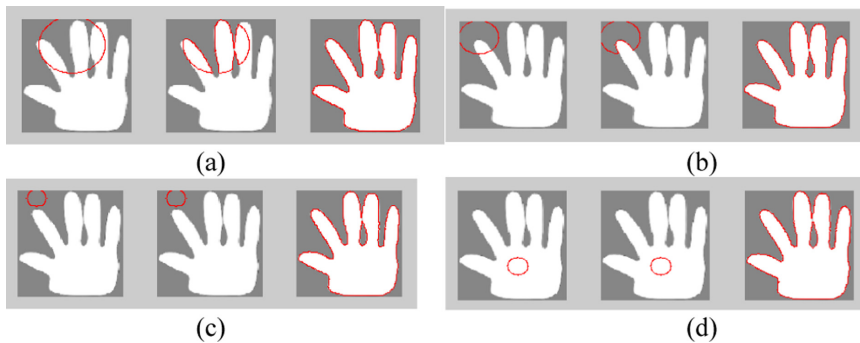


Fig. 2. Robustness to contour initialization. (a), (b) initial contours are intersection with object; (c) initial contour is outside the object; (d) initial contour is inside the object.

In the second experiment, we show its robustness to the parameters and the placement of the initial contour. Fig. 2 shows the proposed algorithm converges to the same object’s boundaries with different initial locations and radius of the contours (intersection or not). Moreover, the experiments also show that the rapidity of

convergence is not affected by initialization. The convergence is achieved in two sweeps in all cases.



Fig. 3. Segmentation results on real images with slight complex background. The first column is given images; the second column is the initial contours; the third column is the results of the first iteration of sweeping; the forth column is results of the final segmentation.

In the third experiment, some real images are used to test the effectiveness of the proposed algorithm. Fig. 3 demonstrates the result of the segmentations, we can clearly see that the sweeping GCF algorithm can segment the real images with slight complex background effectively. As the real images are more complex than the synthetic images used above, more iterations are used to obtain the right segmentation results.

In the fourth experiment, several classical and representative level set methods are adopted to make a comparison. We shall give two real images segmentation based on CV model and GCF [11] based on the gradient descent and finite difference method [3], sweeping algorithm CV [17], and the sweeping algorithm. The parameters of CV model are setting as $\lambda_1 = \lambda_2 = 255$, $\varepsilon = 1$, $u = 1$. The parameters of GCF model are setting as

$\lambda_1 = \lambda_2 = 255, \varepsilon = 1$. The parameters of sweeping CV model are $\lambda_1 = \lambda_2 = 255, \varepsilon = 1$. The same initial contours are used on the test images in all the methods. Segmentation results are shown in Fig. 4 and the compute costs are listed in Table 1. We can see that the CV model and GCF model cannot converge to the right results after 50 iterations. The proposed sweeping GCF model can get better image segmentation and used almost the same time as the sweeping CV model, but much less time than CV model and GCF model. The parameter setting of GCF is rather troublesome than CV model, which requires iteration for many times to get the optimal result, as described in [11].

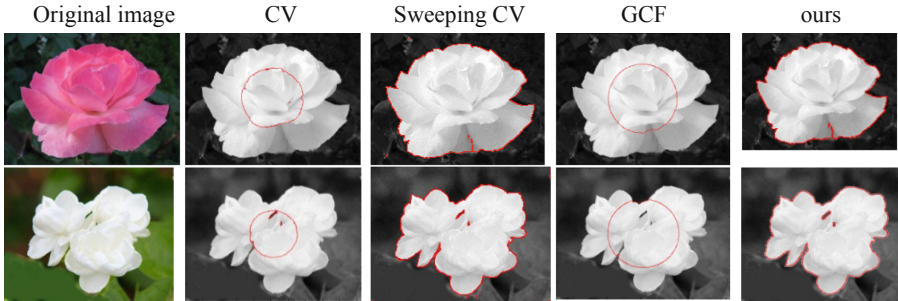


Fig. 4. Segmentation comparisons of CV model, sweeping CV model, GCF model and the proposed sweeping GCF algorithm on real images. The first column: given images; the second column: final segmentation results of CV model with 50 iterations; the third column: final segmentation results of sweeping CV model when the stop condition is meet (the energy is unchanged); the forth column: final segmentation results of GCF model after 50 iterations; the fifth column: final segmentation results of the proposed sweeping GCF algorithm.

Table1. Iterations and CPU time of the experiment in Fig. 4

	CV	Sweeping CV	GCF	ours
	Iternations/Time	Iternations/Time	Iternations/Time	Iternations/Time
Chinese rose 600 * 800 pixs	50/32.235087 s	4/8.570009 s	50/22.333627 s	4/9.377438 s
Jasmine 800 * 1000 pixs	50/240.690795 s	2/15.128192 s	50/37.330821 s	4/16.046365 s

In the last experiment, it was to demonstrate that the proposed algorithm is more robust to noise than the sweeping CV algorithm. The image contains Gaussian white noise with a variance of 0.2% and Speckle noise with a variance of 2%. In Fig. 5, the first row shows the segmentation results by sweeping CV algorithm on the mixture noise image. Four iterations were carried out and the evolving contours of segmentation are showed as follows, the evolution time is 0.325511 s. The results in Fig. 5 show that the CV model cannot remove all of the background noise. The second row shows the segmentation results by the proposed sweeping GCF algorithm. Four iterations are also carried out and the evolving contours of segmentation are showed as in

Fig. 5. The sweeping GCF algorithm can get better segment performance result with the evolution time is 0.419526 s. It can be seen from the experimental results that the sweeping GCF algorithm is more robust to noise image than the sweeping CV algorithm, although the sweeping GCF algorithm takes a little more time. The successful segmentation of our method should owe to the usage of global cosine energy function. Compared with Euclidean distance energy function curve, the energy curve of cosine function is smoother at the bottom, which makes it more robust to noise images.

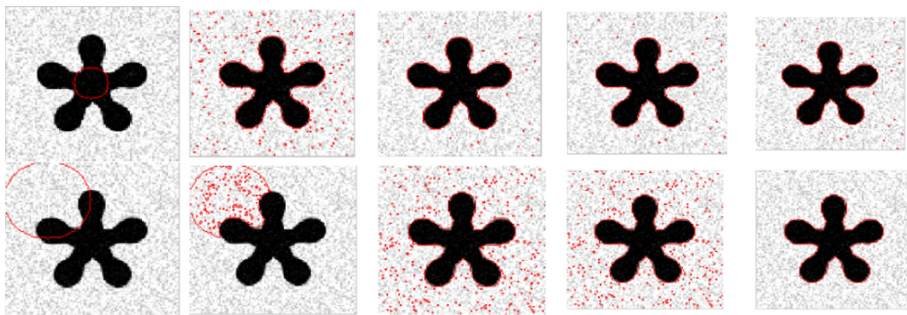


Fig. 5. Comparison of the robust to noise of sweeping CV and the sweeping GCF. The first row: the noise image with evolving contours of each iteration conducted by sweeping CV algorithm. The second row: the noise image with evolving contours of each iteration conducted by sweeping GCF algorithm.

5 Conclusion

Although the level set image segmentation model has been widely used for decades, it remains a popular research topic. In this paper, for global cosine energy general function models, we describe a sweeping optimization algorithm. The algorithm utilizes the sweeping optimization criterion to evolve the level set function. The algorithm does not require solve the energy functional Euler-Lagrange equations and partial differential equations, greatly speeds up the segmentation speed. Compared with traditional level set methods, the proposed sweeping algorithm has the advantages of fast segmentation speed, few iterations, no need to consider Courant-Friendrachs-Lewy conditions, no re-initialization steps, parameter adjustment, algorithm automatic termination of evolution. The proposed algorithm can be more easily extended to high-dimensional image segmentation model.

Moreover, the proposed algorithm can be extensively applied to solve a wider range of optimization problems for the variational image processing problem, where the energy function is easy to compute. But the proposed algorithm is poor for complex background image segmentation and less effective for texture images, strongly noisy images segmentation. We will consider extending the proposed algorithm to multi-phase segmentation, strongly noisy images, textured images, and images with complex backgrounds cases in future.

Acknowledgements. The authors would like to express their thanks to the referees for their valuable suggestions. This work was supported by the grant from the National Natural Science Foundation of China, Nos. 61672204 and 61806068, the Natural Science Foundation of Anhui Provincial, No. 1908085MF184, 1908085QF285, in part by the Key Research Plan of Anhui Province, No. 201904d07020002.

References

1. Gong, X.-Y., Su, H., Xu, De., Zhang, Z.-T., Shen, F., Yang, H.-B.: An overview of contour detection approaches. *Int. J. Autom. Comput.* **15**(6), 656–672 (2018). <https://doi.org/10.1007/s11633-018-1117-z>
2. Malladi, R., Sethian, J.A., Vemuri, B.C.: Shape modeling with front propagation: a level set approach. *IEEE Trans. Pattern Anal. Mach. Intell.* **17**(2), 158–175 (1995)
3. Chan, T.F., Vese, L.A.: Active contours without edges. *IEEE Trans. Image Process.* **10**(2), 266–277 (2001)
4. Wang, X.F., Zou, L., Xu, L.X., Lv, G., Tang, C.: Hybrid level set method based on image diffusion. *Neurocomputing* **228**, 53–64 (2017)
5. Li, C., Xu, C., Gui, C., Fox, M.D.: Distance regularized level set evolution and its application to image segmentation. *IEEE Trans. Image Process.* **19**(12), 3243–3254 (2010)
6. Li, C.M., Kao, C.Y., Gore, J.C., Ding, Z.H.: Minimization of region-scalable fitting energy for image segmentation. *IEEE Trans. Image Process.* **17**, 1940–1949 (2008)
7. Wang, X.F., Huang, D.S., Xu, H.: An efficient local Chan-Vese model for image segmentation. *Pattern Recogn.* **43**(3), 603–618 (2010)
8. Zhang, K.H., Song, H.H., Zhang, L.: Active contours driven by local image fitting energy. *Pattern Recogn.* **43**(4), 1199–1206 (2010)
9. Li, C.M., Huang, R., Ding, Z.H., Gatenby, J.C., Metaxas, D.N., Gore, J.C.: A level set method for image segmentation in the presence of intensity inhomogeneities with application to MRI. *IEEE Trans. Image Process.* **20**(7), 2007–2016 (2011)
10. Min, H., Lu, J., Jia, W., Zhao, Y., Luo, Y.: An effective local regional model based on salient fitting for image segmentation. *Neurocomputing* **311**, 245–259 (2018)
11. Wang, Y., Huang, T.Z., Wang, H.: Region-based active contours with cosine fitting energy for image segmentation. *J. Opt. Soc. Am. A Opt. Image Sci. Vis.* **32**(11), 2237–2246 (2015)
12. Adalsteinsson, D., Sethian, J.A.: A fast level set method for propagating interfaces. *J. Comput. Phys.* **118**, 269–277 (1995)
13. Sethian, J.A.: A fast marching level set method for monotonically advancing fronts. *Proc. Natl. Acad. Sci. U.S.A.* **93**, 1591–1595 (1996)
14. Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Fast geodesic active contours. *IEEE Trans. Image Process.* **10**, 1467–1475 (2001)
15. Yang, Y., Liu, X.: A robust semi-supervised learning approach via mixture of label information. *Pattern Recogn. Lett.* **68**, 15–21 (2015)
16. Wang, X.F., Min, H., Zou, L., Zhang, Y.G., Tang, Y.Y., Chen, C.L.P.: An efficient level set method based on multi-scale image segmentation and hermite differential operator. *Neurocomputing* **188**, 90–101 (2016)
17. Song, B., Chan, T.: A Fast Algorithm for Level Set Based Optimization, CAM-UCLA 68 , pp. 02–68 (2002)
18. He, L., Osher, S.: Solving the Chan-Vese model by a multiphase level set algorithm based on the topological derivative. In: *International Conference on Scale Space and Variational Methods in Computer Vision*, pp. 777–788 (2007)

19. Zou, J.Y., Ma, Y.C.: Pre-sweeping algorithm for solving the Chan-Vese model in image segmentation. *Chin. J. Eng. Math.* **27**(6), 1096–1104 (2010)
20. Krinidis, S., Chatzis, V.: Fuzzy energy-based active contours. *IEEE Trans. Image Process.* **18**(12), 2747–2755 (2009)
21. Shyu, K.K., Pham, V.T., Tran, T.T., et al.: Global and local fuzzy energy-based active contours for image segmentation. *Nonlinear Dyn.* **67**(2), 1559–1578 (2011)
22. Shyu, K.K., Tran, T.T., Pham, V.T., et al.: Fuzzy distribution fitting energy-based active contours for image segmentation. *Nonlinear Dyn.* **69**(1–2), 295–312 (2012)
23. Boutiche, Y.: Fast algorithm to minimize model combining dynamically local and global fitting energy for image segmentation. In: *International Conference on Control, Engineering & Information Technology*, pp. 1–6. IEEE (2015)
24. Boutiche, Y., Abdesselam, A.: Fast algorithm for hybrid region-based active contours optimisation. *IET Image Proc.* **11**(3), 200–209 (2017)
25. Zou, L., et al.: A fast algorithm for image segmentation based on local Chan Vese model. In: In: Huang, D.S., Jo, K.H., Zhang, X.L. (eds.) *Intelligent Computing Theories and Application. International Conference on Intelligent Computing*, pp. 54–60. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95933-7_7
26. Zou, L., et al.: Image segmentation based on Local Chan Vese model by employing cosine fitting energy. In: Lai, J.-H., et al. (eds.) *PRCV 2018. LNCS*, vol. 11256, pp. 466–478. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03398-9_40
27. Boutiche, Y., Abdessalem, A., Ramou, N., Chetih, N.: Fast generalized Chan-Vese model for plant/soil segmentation to estimate percentage of ground cover in agricultural images. In: *2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pp. 1–5 (2019)



Household Garbage Classification: A Transfer Learning Based Method and a Benchmark

Qian Cheng¹, Zhi-Ze Wu³, Zi-Jun Wu³, Le Zou², Huan-Yi Li¹,
and Xiao-Feng Wang²(✉)

¹ Department of Environmental Engineering,
Hefei University, Hefei 230601, Anhui, China

² Anhui Provincial Engineering Laboratory of Big Data Technology Application
for Urban Infrastructure, Department of Computer Science and Technology,
Hefei University, Hefei 230601, China
xfwang@hfu.edu.cn

³ Institute of Applied Optimization, School of Artificial Intelligence and Big
Data, Hefei University, Hefei 23060, Anhui, China

Abstract. Household garbage images are highly diverse in color, texture and geometry, which poses significant challenges to garbage classification. Deep convolutional neural network (DCNN) have recently achieved remarkable progress due to their ability to learn high-level feature representations. It usually requires a large number of labelled image data for training a DCNN model. However, there are few public and mature data sets concerned on household garbage images. This severely limits the progress of research and the state of the art is not entirely clear. To address this problem, we introduce a new benchmark data set for household garbage image classification. This data set is called 30 Types of Household Garbage Images (HGI-30), which contains 6'000 images of 30 household garbage types, with complex backgrounds, different resolutions, and complicated variations in sample, pose, illumination and background. The publicly available HGI-30 data set allows researchers to develop more accurate and robust methods for both household garbage image processing and interpretation analysis of household garbage object. We further study the classification problem on this data set and propose a transfer learning based method, also provide a performance analysis, which serves as baseline result on this benchmark.

Keywords: Household garbage · Benchmark · CNN · Transfer learning

1 Introduction

In recent years, Urban household garbage is growing at an alarming rate. It is very important to recycle those garbage in modern society. In the process of sustainable economic development, effective waste management and recycling are essential. In order to recycle safely and efficiently, we need to rely on intelligent image classification systems instead of employing a large number of workers to do this.

Garbage sorting in the real world is a very challenging computer vision task. As a deformable object, garbage can be transformed into various shapes in various scenes.

For example, garbage can be turned into cardboard of all shapes, bottles of all shapes. In this case, the garbage does not lose its physical properties as garbage, but it does lose some of the key and necessary detection properties of the garbage identification system. In addition, almost all materials can be used as inputs to the garbage classification system, but the number of samples we can train is limited. This requires the system to have good generalization performance when training with a relatively small training set.

Shallow level machine learning mainly has one or two nonlinear feature layers, such as support vector machine (SVM) [1], k-mean clustering [2], logistic regression [3] and gaussian mixture model (GMM) [4]. By using this method, we can classify the garbage by the spectrum, texture and other low-level visual features of the object.

However, these methods are not suitable for dealing with complex issues. Some of the more complex problems such as the problematic image processing these methods are not able to do.

Development of deep learning in recent years have made a breakthrough in these aspects of research [5]. As a form of machine learning, Deep learning [6] has gradually evolved from shallow learning to deep learning.

As a new image classification method, CNN has been introduced [7–9] with the gradual development of deep learning. Krizhevsky et al. [7] introduced the AlexNet network and further developed CNN. Subsequently, after AlexNet achieved good results, some excellent architectures were successively introduced, such as VGG-Net [9], GoogLeNet [10], ResNet [8], etc.

Such networks greatly mitigate the problems of deformation, occlusion, background clutter, which have deep impact on the performance of image classification. However, the advanced CNNs cannot achieve better performance without a lot of training data. Therefore, the problem of a lack of training data is urgent to be solved.

Furthermore, there are few public and mature datasets on household garbage images. This severely limits the deep models based research.

Based on the idea of transfer learning [11], we propose a simple but effective approach for household garbage classification. The pre-trained models on ImageNet dataset are first investigated. In addition to the different model architectures, we also consider using different training parameter ratios to study the influence on the experiment. For the limitation of the image data, we built a new data set, 30 kinds of Household Garbage Images (HGI-30).

There are our contribution:

1. We propose a simple but effective transfer learning based approach for household garbage classification.
2. We construct HGI-30 – the new public household garbage image database. By doing so, we significantly aid the development of robust methods for garbage classification.
3. We describe in detail how this data set is created by applying image simulation methods supported by data gathered from online sources. This allows for applying our approach to generate high-quality benchmark image data sets for other application areas.

The rest of this article is organized as follows. The latest progress of garbage classification is introduced first. In Sect. 3, we set up the HGI-30 data set and gave the

methods and details. In Sect. 4, the details of our experiment are mentioned. There are the results of using different approaches in the new data set we created ourselves, as well as the effect of fine-tuning the ratio of parameters. Finally, conclusions and future work are drawn in Sect. 5.

2 Related Works

One of the early research results is intelligent garbage classification based on Bayesian framework, which is a classic pattern recognition method. Liu *et al.* used Bayesian Classification to implement Material classification. In the dataset, they used Color, SIFT, micro texture and outline shape features [12]. Because it requires manual feature extraction, this method cannot be perfectly automated, despite an excellent mathematical background.

The latest effort by the TechCrunch Disrupt Hackathon team is to create an automated trash can. The purpose of this process is simply to sort the waste [13]. Another project related to environmental resource recovery is the use of smartphone applications by utilizing imaging method [14] to classify garbage. They obtained the data set through Bing image search and trained it using the Alexnet model. The accuracy rate after the training stage was about 87.69%.

A garbage image dataset named “TrashNet” was made which consist of cardboard, plastic, paper, glass, metal and other rubbish. Each type of garbage has about 400 images. Thung et al. extracted SIFT features from the images and then used the CNN structure to classify the images. In this study, they used different parts of classifiers for comparative analysis and adopted a variety of fine-tuning models [15].

Based on TrashNet dataset, Cenk et al. experimented with some CNN models, such as scratch and fine-tuning models. In the scratch model, the Inception-Resnet model achieved the highest classification with a test accuracy of 90% [16]. Umut et al. used two different classifiers to test the performance [17]. Based on the fine-tuning model. They used Support Vector Machine and Softmax.

In this paper, we created a data set, HGI-30, which contains 6000 household garbage images. It was made up of 30 types of garbage. And a method of household garbage classification based on transfer learning is proposed. For VGG16, InceptionV3 [18] and Resnet50, three pre-training models, we design different training parameter ratios to research the classification effect.

3 Materials and Method

3.1 Review of the CNNs

For image classification task, deep convolutional neural network (DCNN) is significant. DCNN is one of the core algorithms of deep learning. It mainly consists of convolutional layer, max-pooling layer, fully-connected layer and softmax classification layer. The original image features are extracted by alternating stacking of

convolutional layer and pooling layer to obtain the generalized abstract representation of the image. The convolutional layer is described in detail below.

A three-dimensional vector of size $W \times H \times N$ is the input of the convolutional layer, with N two-dimensional feature maps X_i^{l-1} ($i = 1, \dots, N$). A three-dimensional vector $W' \times H' \times M$ is the output of this layer, which is composed of M feature maps X_j^l ($j = 1, \dots, M$). M learnable kernels k_{ij}^l (also called weights or filters) of size $F \times F \times N$ are convolved with the input feature maps.

Then, through the nonlinear activation function, the output feature maps X_j^l ($j = 1, \dots, M$) becomes. Therefore, each feature map of the convolutional layer is calculated as

$$X_j^l = f \left(\sum_{i=1}^N X_i^{l-1} * k_{ij}^l + b_j^l \right) \quad (1)$$

where $*$ is the two-dimensional discrete convolutional operator and b_j^l denotes the learnable bias parameter of the j -th output feature map.

Multiple filters can work together in a certain layer at the same time, due to the principle of weight-sharing. The quantity of parameters is only related to the type of filter. The efficiency of feature extraction will be improved while the complexity of the model reduced. Each filter is responsible for extracting a certain feature on the input image. At one time, it identifies only a small area of the image, which is passed to the next convolutional layer. Therefore, the low-level features tend to be abstract and local. The receptive field of the convolution kernel gradually expands, as the level becomes deeper and deeper. The high-level features become more specific and global.

3.2 Transfer Learning

As a machine learning method, transfer learning takes the task A model as the initial position and reuses it in the process of developing the task B model. The feature extraction capability of task A model can be fully released and utilized.

Most CNN architecture consists of approximately tens of millions of parameters. Using random parameter initializations to train such large parameters directly from thousands of training image data can be problematic [19]. Therefore, we used part of the pre-trained CNN model that appeared in the ImageNet Large Scale Visual Recognition Challenge in this work. The ImageNet data set is widely used as a training set in various image classification studies. It divides these large number of images into 1000 object categories. There are more than 1.2 million images. Then, We transfer these pre-training models to a new model to better fit the special task.

For the full connection layer of these models, the old one will be replaced by the new one. Since our task-specific dataset HGI-30 contains 30 types of household garbage, the output vector dimension becomes 30.

3.3 The Fine-Tuning Stage

For a new data set, we adjust the model parameters to make a pre-trained DCNN model with specific parameters adapt to the new task, which is the idea of fine-tuning. For a

typical DCNNs, the features of the low-level are generally extracted from the previous layer. Basic features like lines, edges, etc. The top layers are the advanced features that are relevant to the new task. Therefore, the low-level parameters of the pre-training model can be transferred. The parameters of the upper-level model need to adapt to the specific task in the fine-tuning stage.

In this research, we designed different training parameter ratios to control the amount of the update parameters. By freezing different training layers, we explored the influence of different training parameter ratios for our specific image classification task.

3.4 Benchmark

In order to better study the garbage image classification field and evaluate the new method proposed by us, a new data set was established. The new household garbage image classification dataset HGI-30 contains 6,000 images. It contains 30 types of household garbage. We capture these images through photography.

(1) Sample Variations:

The change of sample is an essential factor that affects the detection of garbage image significance. Different samples will change in real applications. To assess its effectiveness, we considered four different samples of each type of garbage, as shown in Fig. 1.



Fig. 1. The garbage images captured with four different sample variations

(2) Pose Variations:.

To capture a garbage image of an object in different poses, we considered four sides (left, right, front and back) of a garbage object. We use the camera to take a garbage image on each side. For each garbage object, we get four garbage images. Figure 2. shows an example of the four directions for each garbage object.



Fig. 2. Two typical garbage objects with diverse postures: (a) left-, (b) right-, (c) front-, and (d) opposite-, for each garbage object.

(3) Background Variations:

While it is nice to see that significant progress has been made by many image recognition methods at present, their performance is still unsatisfactory in some special cases, such as complex image background. Therefore, background variations are very important for the performance evaluation of image database and garbage classification method. So, for the HGI-30 data set, we consider three background changes. Figure 3 shows some sample garbage images under three different backgrounds.



Fig. 3. Garbage objects at different background variations.

(4) Illumination Variations:

In the process of garbage image capture, we chose three different lighting conditions. Figure 4. shows the sample garbage images under three different illumination conditions.

There are 6000 garbage images in this data set, all in RGB color format. For the original image, the resolution per image is up to 2048×1536 pixels. The storage capacity required for the entire database is approximately 19.1 GB, which we consider being a conundrum for researchers using this data set. The proportional scaling with a down-sample factor $1/4$ is something that we have taken into account to mitigate this problem, and doing so still retains most of the vital information. Therefore, all the images were processed by us and reduced to 512×384 pixels. When this is done, each original image resolution becomes a quarter of the original image resolution.



Fig. 4. The garbage images captured with three different lighting variations.

4 Experiments and Results

In this research, Keras [20] library (Version 2.2.4) with TensorFlow backend was used on Window10 Professional platform. In the experiment, GTX1080 was used with Intel Core i7–8700 CPU at 3.2 GHz and 16 GB memory.

For VGG16, InceptionV3 and Resnet50, three pre-training models, we fine-tuned the weight of those models. We set a number for the learning rate, which is 0,0001. Stochastic gradient descent with 0.9 Nesterov momentum was used by us. Set the batch size to 32. In addition to studying different model architectures, the impact of different training parameter ratios was also considered.

It is divided into two groups of image data. Seventy percent of this data is used for training and 30 percent for validation. For the data we built, we also used several different data augmentation methods, including vertical flips and 15-degree random rotation. The purpose of this is to make the generalization ability of neural networks all improve.

Table 1 shows the results in the comparison experiments. Figure 5 show accuracy for different training parameter ratios on VGG16, InceptionV3 and Resnet50.

The experimental results show that this transfer model based on neural network can achieve better classification accuracy. In the fine-tuning experiment of 6000 image data sets, all the DCNN-based models we tried could reach more than 90% accuracy in the

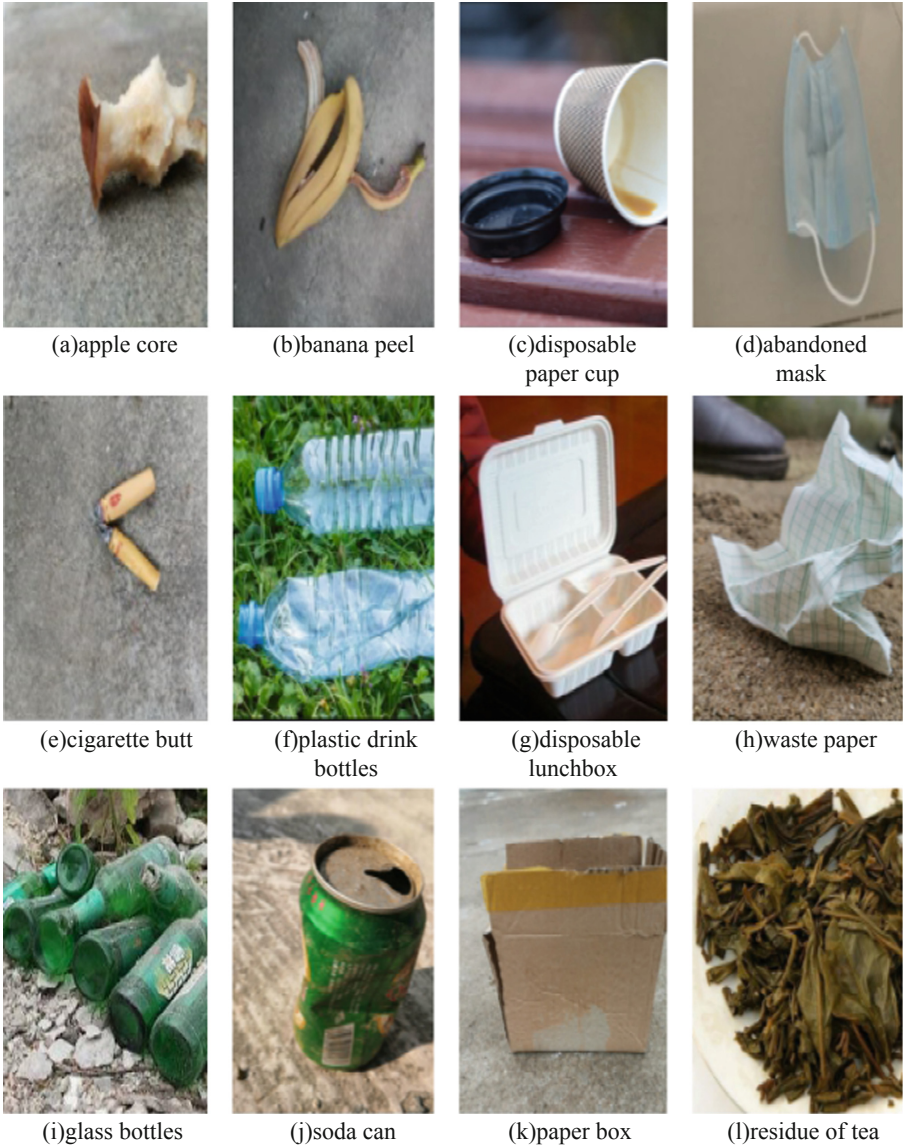


Fig. 5. Some sample images of HGI-30 dataset.

test. For some models, such as EfficientNetB2, it scored a high 96.6%. Compared with the traditional classification method, our proposed method has more advantages.

With using appropriate training parameter ratio, we achieved excellent performance improvement of the pre-training model. In our experiment, Inceptionv3 model has the highest test accuracy, reaching 97.5% when the training parameters account for 75% (Fig. 6).

Table 1. The performance comparison of different household garbage classification methods on HGI-30. As far as test accuracy is concerned, the bigger the better. Note that “ours” is to train InceptionV3 with 75% of the training parameter ratio.

Method	Test accuracy(%)	Data aug
SIFT [21] + BOVW	64.5	+
HOG [22] + SVM	67.8	+
VGG16	90.5	-
VGG16	91.3	+
ResNet50	92.6	-
ResNet50	94.5	+
InceptionV3	93.5	-
InceptionV3	95.1	+
Mobilenet [23]	92.3	+
DenseNet121 [24]	96.2	+
Xception [25]	95.5	+
EfficientNetB0 [26]	96.1	+
EfficientNetB1 [26]	96.4	+
EfficientNetB2 [26]	96.6	+
Ours	97.5	+



Fig. 6. Accuracy for different training parameter Ratios.

5 Conclusions and Future Work

The classification of household garbage images attracts much research interest. In this paper, we first review several recent work on the household garbage classification. We found that all of them either are not suitable for our domain or have significant shortcomings. This inspired us to develop a new benchmark data set for household garbage classification, HGI-30.

On the new data set HGI-30, transfer learning was used on the DCNN architecture to achieve the highest accuracy. We also investigate different training parameter ratios for studying the performance of transfer learning. For this limited dataset of household garbage, we obtained 97.5% test accuracy.

From the experimental results, we argue that the transfer learning based intelligent system could effectively perform garbage classification task. It is also a promising example of the use of artificial intelligence or more specifically deep learning on behalf of ecological consciousness.

In the future, we plan to experiment with different optimization algorithms and different classifiers. We will enrich and perfect our proposed HGI-30 data set, and then make it available to the public.

Acknowledgements. This work was supported by the Youth Project of the Provincial Natural Science Foundation of Anhui Province 1908085QF285 and 1908085QF262, the National Natural Science Foundation of China Grant 61672204, the Anhui Provincial Education Department Project, KJ2019A0834 and KJ2019A0835, the Key Research Plan of Anhui 201904d07020002, the Scientific Research and Development Fund of Hefei University 19ZR05ZDA, the Talent Research Fund Project of Hefei University 18-19RC26.

Disclosure All authors declare that there are no conflicts of interests. All authors declare that they have no significant competing financial, professional or personal interests that might have influenced the performance or presentation of the work described in this manuscript. Declarations of interest: none. All authors approve the final article.

References

1. Cortes, C., Vapnik, V.: Support-vector networks. *Machine learning* **20**(3), 273–297 (1995)
2. MCQUEEN, J.: Some methods for classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297 (1967)
3. Kleinbaum, D.G., Dietz, K., Gail, M., et al.: *Logistic regression*. Springer-Verlag, New York (2002)
4. Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. In: *Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004*. vol. 2, pp. 28–31, IEEE (2004)
5. Schmidhuber, J.: Deep learning in neural networks: an overview. *Neural Networks* **61**, 85–117 (2015)
6. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. *Nature* **521**(7553), 436–444 (2015). <https://doi.org/10.1038/nature14539>
7. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: *Proceedings of the Advances in Neural Information Processing Systems*, pp. 1097–1105. Lake Tahoe, NV, USA, 3–6 December 2012
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA* **27–30**, 770–778 (2016)

9. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv 2014, [arXiv:1409.1556](https://arxiv.org/abs/1409.1556).
10. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9. Boston, MA, USA, 7–12 June 2015
11. Dai, W., Chen, Y., Xue, G.R., Yang, Q., Yu, Y.: Translated Learning: Transfer Learning across Different Feature Spaces. *Adv. Neural Inf. Process. Syst.* **21**, 353–360 (2008)
12. Liu, C., Sharan, L., Adelson, E. H., Rosenholtz, R.: Exploring features in a bayesian framework for material recognition. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, pp. 239–246 (2010)
13. Donovan, J.: Auto-trash sorts garbage automatically at the techcrunch disrupt hackathon (2018)
14. Mittal, G., Yagnik, K.B., Garg, M., Krishnan, N.C.: Spotgarbage: Smartphone app to detect garbage using deep learning. In: Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing, ser. UbiComp 2016. pp. 940–945, ACM, New York (2016)
15. Thung, Gary, Yang, M.: Classification of Trash for Recyclability Status (2016)
16. Bircanoglu, C., Atay, M., Beser, F., Genc, O., Kizrak, M.A.: RecycleNet: intelligent waste sorting using deep neural networks. In: 2018 Innovations in Intelligent Systems and Applications (INISTA) <https://doi.org/10.1109/inista.2018.8466276>.
17. Ozkaya, U., Seyfi, L.: Fine-tuning models comparisons on garbage classification for recyclability. *Computer Vision and Pattern Recognition* (2019)
18. Szegedy, C., Vanhoucke, V., Ioffe, S., et al.: Rethinking the inception architecture for computer vision. In: Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition. pp. 2818–2826. IEEE, Piscataway (2016)
19. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1717–1724 (2014)
20. Chollet, F., et al.: Keras (2015)
21. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**, 91–110 (2004)
22. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* **1**, 886–893 (2005)
23. Howard, A.G., Zhu, M., Chen, B., et al.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint [arXiv:1704.04861](https://arxiv.org/abs/1704.04861) (2017)
24. Huang, G., Liu, Z., Van, Der, Maaten, L., et al.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
25. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision And Pattern Recognition, pp. 1251–1258 (2017)
26. Tan, M., Le, Q.V.: Efficientnet: rethinking model scaling for convolutional neural networks. arXiv preprint [arXiv:1905.11946](https://arxiv.org/abs/1905.11946) (2019)



Lightweight Neural Network Based Garbage Image Classification Using a Deep Mutual Learning

Xiao Liu¹, Zhi-Ze Wu³, Zi-Jun Wu³, Le Zou², Li-Xiang Xu²,
and Xiao-Feng Wang²(✉)

- ¹ Department of Environmental Engineering, Hefei University, Hefei 230601, Anhui, China
- ² Anhui Provincial Engineering Laboratory of Big Data Technology Application for Urban Infrastructure, Department of Computer Science and Technology, Hefei University, Hefei 230601, China
xfwang@hfu.edu.cn
- ³ Institute of Applied Optimization, School of Artificial Intelligence and Big Data, Hefei University, Hefei, Anhui 230601, China

Abstract. With the construction and development of civilized cities, image based garbage classification has gradually become an important concern in computer vision community. During the algorithms for image classification, the strong ability of Convolution Neural Networks (CNNs) in feature learning makes it the most successful approach at the moment. However, the parameters of CNNs model are very huge, and its training usually depends on a large amount of samples. In this article, we tackle the problem of lightweight neural network based garbage image classification, which aims to learn classifier with a small number of model parameters. Specifically, we utilize the MobileNetV2 for the backbone of feature extraction network and jointly train such two nets in a way of deep mutual learning. It realizes the information distillation between the teacher and the student. With this, we can significantly improve the learning ability of the MobileNetV2 based lightweight neural network. The experimental results on a self-assembled dataset show that our proposal effectively classifies the garbage and achieves a classification effect better than the state of the arts in terms of testing accuracy, time and model size.

Keywords: Garbage classification · Lightweight neural network · Deep mutual learning · Distillation

1 Introduction

Urban domestic waste is increasing at an annual rate of 5%–8%. Many cities are facing the problem of “garbage siege” [1]. A large amount of garbage is randomly discarded without classification, which causes environmental pollution and affects people’s living standards. Therefore, it is urgent to solve the garbage classification problem. At present, the domestic garbage treatment mainly relies on manual sorting, which is unfavorable to the health of the staff. Moreover, the manual sorting efficiency is low, which

cannot meet the demand for large amounts of garbage. In addition, the types of manually sorted garbage are extremely limited, and most garbage cannot be recycled, resulting in a huge waste of resources. With the development of computer vision technology, we argue the possibility of automatically sorting and sorting garbage. It can greatly reduce labor costs and improve sorting efficiency. Therefore, it is valuable for studying the garbage image classification by combing the modern successful computer vision algorithms.

Early years, there were many major achievements in computer vision technology, one of which was feature extraction combined with classifiers to classify images. For example, Ma et al. [2] proposed the research of pipeline inner surface image classification algorithm based on support vector machine and distance measurement; Li et al. [3] proposed the SVM learning strategy based on the improved LBG algorithm; In [4], Ren et al. proposed Research on image scene classification based on LDA topic model. However, these traditional image processing methods require manual feature extraction and specific methods for specific problems. If the features are not properly selected, the classification performance and the model robustness will be poor.

Deep learning is a relatively new artificial intelligence technology. Convolutional neural networks (CNNs), with their powerful learning and feature expression abilities, have achieved better results in many applications of computer vision compared with traditional methods. Recently, deep learning based methods are constantly studied for image classification task. These methods can be mainly divided into two types, namely deep networks and lightweight networks. Deep models are the standard network model, such as LeNet5 [5] proposed by Yann LeCun et al., AlexNet [6] proposed by Krizhevsky et al., and later VGG [6], ResNet [7], GoogleNet [8]. Except the deep models, lightweight neural network also attracts much research interest. As we know, MobileNet [9] is the most representative one.

However, the above neural network structure has limitations, such as the inability to balance the parameters and learning ability. For example, the VGG16 parameters are too large and the model size occupies 500 MB, so it requires more computer resources and cannot be used on general machines. The lightweight neural network MobileNet has fewer parameters and consumes less resources, but its learning ability is limited. In response to the above two issues, we propose a lightweight neural network based approach, which utilizes the MobileNetV2 [10] as a backbone for feature extraction network, and combines with deep mutual learning [11]. By introducing the mutual learning, we can realize the knowledge transfer from the teacher model to the student model at a low memory and a fast execution. As a result, a simple student network learns from each other and supervises each other to have good performance ability [12].

This model performs surprisingly on a garbage dataset composed of 12 common garbage and a total of 12,000 garbage pictures. The results show that the model used in this paper not only greatly reduces the number of parameters of the neural network, the training speed is 1/5 of VGG19, but also improves the accuracy of the MobileNetV2 by 2.8%.

2 Related Work

In 2016, Yang et al. constructed the TrashNet Dataset for public garbage classification, which contains 6 categories and a total of 2,527 images, including 403 cardboard, 501 glass, 410 metal, and 594 paper. 482 pieces of plastic and 137 pieces of other waste materials. Mindy Yang conducted preliminary experiments on this dataset and achieved 63% accuracy with the SVM method [13]. Later researchers quickly followed up. For example, Stephenn L. Rabano used the lightweight neural network MobileNet and achieved a test accuracy of 87.2% [14]. Ozkaya compared a variety of classification and extraction feature networks with classifiers and found that GoogleNet with SVM classifier has the best effect on the TrashNet Dataset so far [15], with an accuracy of 97.86%. In addition, Gaurav Mittal self-made GINI, which is a non-public dataset containing 2561 spam images, and used the GarbNet model to obtain an accuracy of 87.69% [16]. There are few domestic researches in the field of garbage classification. Wu Jian used traditional computer vision methods to manually extract color and texture features in 2016 to achieve the separation of garbage and background in laboratory scenes [17]. Xiang Wei used the improved CaffeNet to recognize water surface garbage with an accuracy of 95.75% [18]. Zheng Hailong used the SVM method to conduct research on the classification of construction waste [19]. In 2019, Huawei organized a garbage image classification competition and constructed dataset with more than 10,000 samples, which further promoted the development of this field.

3 Innovation

The classification standards of domestic waste in various regions of our country are different. They can be roughly divided into four categories: kitchen waste, recyclable waste, hazardous waste, and other waste. Each category contains several sub-categories, which are numerous and complex. Domestic research on garbage identification and classification is still in its infancy, and existing classification algorithms are rarely used in this field. Based on a large number of relevant researches at home and abroad, this topic proposes a classification network structure used on a lightweight neural network MobileNetV2 based on deep mutual learning for the current problems in garbage image classification. The innovative work of this paper has the following aspects:

A relatively representative garbage image dataset was created, including 4 categories, 12 sub-categories, and 12,000 images.

In-depth study of the deep learning models MobileNet V1, MobileNet V2, and deep mutual learning model, and combined them. In addition, made several versions of improvements: model structure, optimizer and learning rate adjustment strategy. In the experiment, the model was visually tested, and the reasoning process and performance of the model were evaluated more deeply.

Propose a lightweight neural network DML_MobileNetV2 model architecture based on deep mutual learning, which significantly reduces model capacity, improves model calculation speed, can be better applied in embedded devices, and overcomes disadvantages such as low precision.




4 Experimental Dataset and Preprocessing

In deep learning, the construction of a dataset is very important, which is the basis for subsequent experiments and research. We used 12 kinds of common garbage datasets as the original datasets for this experiment, and did data preprocessing and image enhancement.

4.1 Dataset Architecture

Garbage is divided into kitchen waste, recyclable garbage, hazardous garbage, and other garbage. We chose three common garbage from each category, a total of 12 categories, and 4500 pictures as the original dataset of this experiment (Table 1).

Table 1. Garbage and the number of samples of each species

			
(1) Battery 350	(2) Ointment 150	(3) Light tube 100	(4) Beef 600
			
(5) Egg 500	(6) Leftovers 350	(7) Pencil 320	(8) Bag 710
			
(9) Mobile phone 600	(10) Face mask 320	(11) Power bank 350	(12) Ticket 150

4.2 Dataset Preprocessing and Data Enhancement

The source of the dataset used in this article is from camera collection and web image crawling, with different image sizes. Therefore, it is necessary to cut the image into $224 * 224$ to meet the neural network model's requirements for the input data format.

The original dataset is not uniformly distributed and the total sample size is too small, which may easily cause over-fitting. In this experiment, we needed to enhance the dataset; each kind of dataset can be enhanced to about 1000, and the total number of samples is about 12000. At the same time, the dataset is divided into training set and test set at a 9:1 ratio.

5 Garbage Classification Model Architecture

In view of the shortcomings of common neural networks, this paper proposes to use two MobileNetV2 models, combined with deep mutual learning methods, so that these two lightweight models can achieve high efficiency and precision. It can effectively solve the problems of large memory requirements and weak learning ability of common neural networks.

5.1 MobileNetV2 Model Architecture

Andrew Howard also proposed an improved version of MobileNetV2 on the basis of MobileNetV1 in 2018. MobileNetV2 is a lightweight model, which is characterized by a small model, fast calculation speed and can be deployed on mobile and embedded devices. It is characterized by the use of deep separable convolution instead of standard convolution, and the introduction of an inverted residual network structure.

Standard Convolution. The input feature matrix of standard convolution is $D_F * D_F$, the size of the convolution kernel is $D_K * D_K$, M is the depth of the input feature matrix, and N is the depth of the output feature matrix (Fig. 1).

Standard convolutions have the computational cost of:

$$D_K * D_K * M * N * D_F * D_F \tag{1}$$

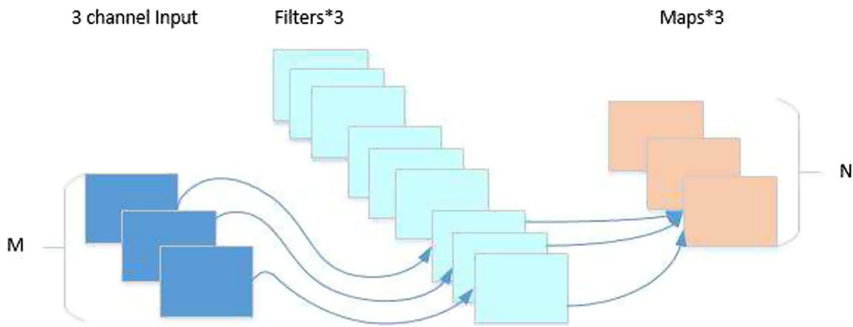


Fig. 1. The calculation process of standard convolution

Depth Separable Convolution. Depth separable convolution is dividing the traditional convolution process into two parts: Depthwise Convolutional Filters and Pointwise Convolutional Filters. Depthwise Convolutional Filter uses each channel of the input feature matrix to correspond to a convolution kernel when calculating convolution. This step only changes the height and width of the input feature matrix, while the number of channels remains unchanged. Pointwise Convolutional Filters use a $1 * 1$ size convolution kernel for convolution, which only changes the number of channels of the feature matrix .

Depth separable convolution have the computational cost of:

$$D_K * D_K * M * D_F * D_F + M * N * D_F * D_F \tag{2}$$

Through calculation, it can be known that the depth separable convolution will reduce the amount of parameters (Fig. 2):

$$\frac{D_K * D_K * M * D_F * D_F + M * N * D_F * D_F}{D_K * D_K * M * N * D_F * D_F} = \frac{1}{N} + \frac{1}{D_K^2} \tag{3}$$

If we take a 3 * 3 convolution kernel, we know that the amount of parameters is approximately reduced by 9 times.

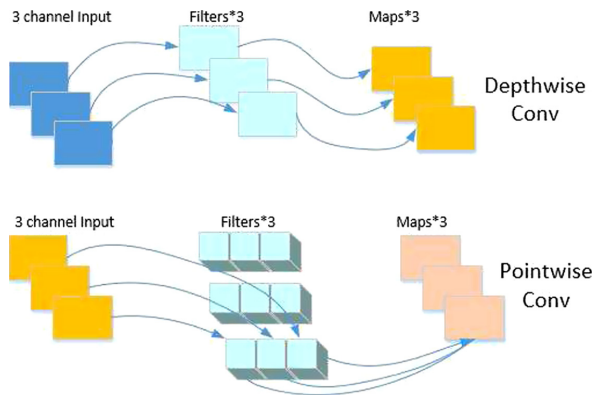


Fig. 2. The calculation process of depth separable convolution

Inverted Residual Structure of MobileNetV2. The inverted residual structure is to first perform 1 * 1 convolution of the input feature matrix to increase the dimension, then perform 3 * 3 convolution, and finally use 1 * 1 again to perform the dimension reduction operation, which can improve the feature extraction ability of the model.

At the same time, the ReLU is replaced with Linear. The features that are less than zero are input to the ReLU function, and the output is all zero. This will cause the features that have been compressed to be compressed again, and the loss is relatively large. Therefore, after the layer with fewer channels, linear activation is used instead of ReLU (Fig. 3).

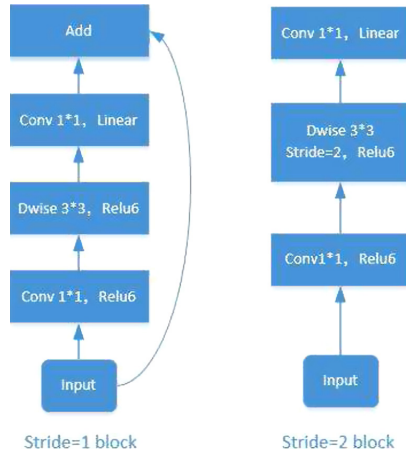


Fig. 3. Inverted residual network structure diagram

5.2 The Experimental Model Architecture

This experiment uses a deep mutual learning model, combined with the MobilenetV2 backbone feature extraction network. The deep mutual learning model is derived from the knowledge distillation model. Two lightweight models learn from each other and guide each other during the training process. With this method, a compact network can be obtained, and a good classification effect can be achieved while greatly reducing the amount of parameters (Fig. 4).

In the deep mutual learning network, in the training process of each backbone feature extraction network, on the one hand, it improves its own prediction accuracy through self-supervised learning; on the other hand, it should fit other network prediction results as much as possible.

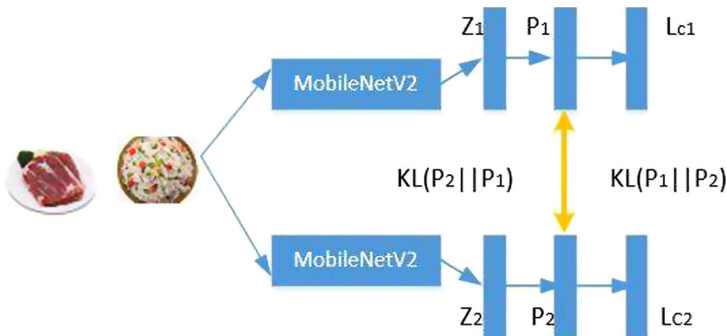


Fig. 4. Schematic diagram of the deep mutual learning model of this experiment

Given N samples of M categories, it is expressed as $X = \{X_i\}_{i=1}^N$, the corresponding label of each sample can be expressed as $Y = \{Y_i\}_{i=1}^N$, where $y_i \in \{1, 2, \dots, M\}$. After the sample x_i is extracted by the MobileNetV2 neural network, it is classified using the Softmax [20] classifier, so the category probability formula generated by the classification is:

$$p_1^m(x_i) = \frac{\exp(z_1^m)}{\sum_{m=1}^M \exp(z_1^m)} \quad (4)$$

For multi-class classification loss, we use the cross-entropy loss function, the formula is:

$$L_{c1} = - \sum_{i=1}^N \sum_{m=1}^M I(y_i, m) \log(p_1^m(x_i)) \quad (5)$$

We use KL divergence to measure the predicted values p_1 and p_2 of the two networks. The KL divergence of p_2 to p_1 is:

$$D_{KL}(P_2||P_1) = \sum_{i=1}^N \sum_{m=1}^M p_2^m(x_i) \log \frac{p_2^m(x_i)}{p_1^m(x_i)} \quad (6)$$

In summary, the total loss function of a MobileNetv2 network is:

$$L = L_{c1} + D_{KL}(P_2||P_1) \quad (7)$$

6 Experimental Results and Analysis

6.1 Experimental Environment and Parameter Settings

This experiment uses the Ubuntu16.04 operating system to build a deep learning environment, a NVIDIA GeForce GTX 1060 graphics card, Python 3.6 for a programming environment, and the framework is PyTorch 1.0.0.

This paper uses the MobileNetV2 model based on deep mutual learning for experimental training. The experimental process is set to 200 iterations, the optimizer is Adam, the initial learning rate parameter is set to 0.001, the learning rate decay is set to gradient decay, and cross-entropy loss is combined KL divergence is used as the loss function of this model.

6.2 Experimental Results

The average accuracy of the test set of the MobileNetV2 composite neural network based on deep mutual learning proposed in this paper is as high as 97.9%, while the

two models are only 28 MB in total. We feed the training set images to the neural network for training, and obtain the loss value of the training set. After each iteration is completed, the test set is fed to the neural network to obtain the accuracy.

The experimental results show that the neural network participating in the comparison experiment, the loss value of the training set and the accuracy value of the test set converge after 20 iterations. Compared with MobileNetV2, DML_MobileNetV2 neural network converges faster, converging within 140 rounds of iteration. The loss value of the training set is 0.10, which is 2% lower than the loss value of the MobileNetV2 training set. Compared with VGG16, DML_MobileNetV2 has more obvious advantages, and the loss value of the training set is 6.2% lower. The training results are shown in Fig. 5.

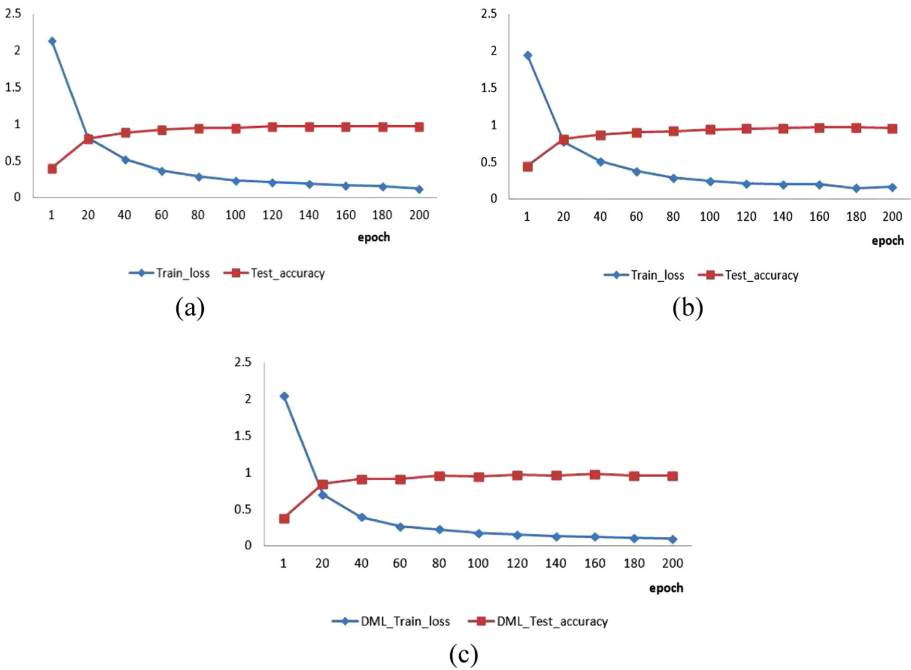


Fig. 5. Single MobileNetV2 (a) VGG16 (b) MobileNetV2 train dataset loss and test dataset accuracy based on deep mutual learning (c)

After the completion of 200 iterations, the average accuracy, recall, and specificity of each type of garbage image are shown in the following Table 2. It can be seen from the table that the highest accuracy rate is eggs and bills, and the lowest is packages. This is because the shape of eggs and bills is relatively simple, and feature recognition is relatively easy, while the shape of the package is rich in colors, which makes feature recognition difficult, which leads to a higher probability of model misjudgment.

Table 2. Confusion matrix representation

	Precision	Recall	Specificity
Bag	0.95	0.98	0.99
Battery	0.97	0.96	0.99
Beef	0.99	0.97	0.99
Egg	1.00	0.99	1.00
Face mask	0.99	0.99	0.99
Leftovers	0.98	0.97	0.99
Light tube	0.99	0.98	0.99
Mobile Phone	0.98	0.97	0.99
Ointment	0.97	1.00	0.99
Pencil	0.98	0.97	0.99
Power Bank	0.96	0.98	0.99
Ticket	1.00	1.00	1.00

Compared with other image classification models, the advantages of the MobileNetV2 model based on deep mutual learning are shown in Table 3 below. The accuracy of the ResNet50 image classification model on the test set is 0.1% higher than that of DML_MobileNetV2, but the size of the model is 3.5 times that of DML_MobileNetV2, which is not suitable for deploying mobile terminals. The size of a single MobileNetV2 image classification neural network model is only half of the DML_MobileNetV2 network, but the accuracy rate on the test set is lower by 2.8%. Other models participating in the comparison experiment, such as AlexNet, VGG16 and VGG19, are far inferior to DML_MobileNetV2 in terms of model size and accuracy of the test set.

Table 3. Compared with other models, the advantages of this model are shown in the following table:

	Model size	Accuracy	Speed/epoch
AlexNet	56 MB	90.2%	49.04 s
VGG16	528 MB	94.8%	272.43 s
VGG19	549 MB	95.0%	315.53 s
ResNet50	99 MB	98.0%	124.13 s
MobileNetV2	14 MB	95.1%	51.25 s
DML_MobileNetV2_1/_2	14 MB/14 M	97.9%/97.9%	62.37 s

7 Conclusion

This article uses a combination of lightweight neural network MobileNetV2 and deep mutual learning. Two MobileNetV2 neural networks do not only need to fit the training data during the training process, but also the prediction results of another network. The two networks learn from each other, which can greatly improve the performance of the network. According to the results, this method effectively improves the weak learning ability of lightweight neural networks, and avoids the problems of too many common neural network parameters and poor transplantation. This paper uses this method to effectively classify common garbage, with a recognition accuracy as high as 97.9%, helping solve the problem of difficult garbage image classification to a certain extent. Additionally, because of its few parameters and fast running speed, it can be embedded in mobile devices, which has a good application prospect in automatic garbage classification.

Due to the large amount of data in the process of camera shooting and web crawling, the original dataset will have inconsistencies between the image content and the label, which will reduce the adaptability of the model. Therefore, compared with the application of deep learning in other areas, the accuracy of the model trained in this article will be further improved after optimization.

Acknowledgements. This work was supported by the Youth Project of the Provincial Natural Science Foundation of Anhui Province 1908085QF285 and 1908085QF262, the National Natural Science Foundation of China Grant 61672204, the Anhui Provincial Education Department Project, KJ2019A0834 and KJ2019A0835, the Key Research Plan of Anhui 201904d07020002, the Scientific Research and Development Fund of Hefei University 19ZR05ZDA, the Talent Research Fund Project of Hefei University 18-19RC26.

Disclosure

All authors declare that there are no conflicts of interests. All authors declare that they have no significant competing financial, professional or personal interests that might have influenced the performance or presentation of the work described in this manuscript. Declarations of interest: none. All authors approve the final article.

References

1. Liu, Q.X.: Research on comprehensive treatment of municipal solid waste. *Smart City* (9), 226–227 (2016)
2. Ma, Y.J., Fang, K., Wang D.C.: Research on pipeline inner surface image classification method based on support vector machine and distance measurement. *Data Acquisition Process*. (02):151–155 (2002)
3. Li, T., Wang J.P., Wu X.Q., Zhang, S.Y.: SVM learning strategy based on improved LBG algorithm. *Fudan J. (Natural Science Edition)*, (05), 789–792 (2004)
4. Ren, Y.: Research on Image Scene Classification Based on LDA Topic Model[D]. North University of China, (2017)
5. LéCun, Y., Bottou, L., Bengio, Y.: Gradient-based learning applied to document recognition. *Proc. IEEE* **86**(11), 2278–2324 (1998)

6. Alex, K., Ilya, S., Geoffrey, E.H.: ImageNet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, pp. 1097–1105 (2012)
7. Simonyan, K., Andrew Z.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556)(2014)
8. He, K., Zhang, X.Y., Ren, S.Q., Jian, S.: Deep residual learning for image recognition arXiv preprint [arXiv:1512.03385](https://arxiv.org/abs/1512.03385) (2015)
9. Andrew, G.H., Zhu, M.L., Chen, B., Dmitry, K.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint [arXiv:1704.04861](https://arxiv.org/abs/1704.04861) (2017)
10. Mark, S., Andrew, H., Zhu, M.: MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv preprint [arXiv:1801.04381](https://arxiv.org/abs/1801.04381) (2019)
11. Zhang, Y., Xiang, T., Timothy, M., Lu, H.C.: Deep Mutual Learning. arXiv preprint [arXiv:1706.00384](https://arxiv.org/abs/1706.00384) (2017)
12. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. *Comput. Sci.* **14**(7), 38–39 (2015)
13. Cortes, C., Vapnik, V.: Support-vector networks. *Mach. Learn.* **20**(3), 273–297 (1995)
14. Rahmi, A.A., Şeref, R.K., Mahmut, K.: Classification of TrashNet dataset based on deep learning models. In: *2018 IEEE International Conference on Big Data*, pp. 2058–2062 (2018)
15. Umut, O., Levent, S.: Fine-tuning models comparisons on garbage classification for recyclability. arXiv:1908.04393 (2019)
16. Mittal, G, Yagnik, K.B., Garg, M.: SpotGarbage: smartphone app to detect garbage using deep learning. In: *2016 ACM International Joint Conference* (2016)
17. Wu, J., Chen, H., F. Wu.: Research on waste and garbage analysis and identification based on computer vision. *Inf. Technol. Informatization* **16**(10), 81–83 (2016)
18. Xiang, W., Shi, J.F., Liu, G.H.: Application of improved CaffeNet model in water surface garbage identification. *Sens. Microsyst. Syst.* **38**(8), 150–156 (2019)
19. Zheng, L.H., Yuan, Z.Q., Yin, C.B.: Research on automatic classification system of construction waste based on machine vision. *Mech. Eng. Autom.* **2019**(6), 16–18 (2019)
20. Liu, W., Wen, Y., Yu, Z.: Large-margin softmax loss for convolutional neural networks. In: *Proceedings of the 33rd International Conference on Machine Learning*, pp. 507–516 (2016)



VBSSR: Variable Bitrate Encoded Video Streaming with Super-Resolution on HPC Education Platform

Run Wu^{1,2}, Gangqiang Zhou^{1,2}, Miao Hu^{1,2}, and Di Wu^{1,2,3}(✉)

¹ Department of Computer Science, Sun Yat-sen University, Guangzhou, China
wudi27@mail.sysu.edu.cn

² Guangdong Key Laboratory of Big Data Analysis and Processing,
Guangzhou, China

³ Peng Cheng Laboratory, Shenzhen, China

Abstract. In online HPC education platforms, a large amount of educational resources are in the form of video. It is desirable to provide better QoE (Quality of Experience) for students when they are viewing these educational resources. Compared to traditional constant bitrate (CBR) encoding, variable bitrate (VBR) encoding can achieve better video quality and reduce network bandwidth. However, previous adaptive bitrate (ABR) schemes were commonly designed for CBR encoded videos. Such ABR schemes are not suitable to stream VBR encoded videos whose chunk sizes fluctuate rapidly. In this paper, we propose a novel ABR scheme call VBSSR, which takes the characteristics of VBR encoded video into consideration. The basic idea of VBSSR is to stream video chunks with complex scenes at a low bitrate level to reduce bandwidth consumption, and then boost the video quality by leveraging the technique of super-resolution (SR) at the client-side. VBSSR trains a deep reinforcement learning (DRL) based neural model to jointly make bitrate selections and decide which chunks to be enhanced. We conduct extensive trace-driven evaluations to compare VBSSR with other state-of-the-art methods. The experiment results show that our method significantly outperforms existing approaches with improvements in the average video quality by at least 37.1% while reducing the rebuffering time by 24.3%–75.9%.

Keywords: Adaptive video streaming · VBR encoding · Super-resolution

1 Introduction

Recent years have witnessed tremendous growth in video streaming market to the fields of research, science, and education. High-quality education and technology have the power to change the world by developing the human talent required to seize the opportunities that arise from global change, e.g., the HPC education

platform [9, 18]. The video streaming technology can be regarded as an enabler and a complementary tool for high-quality HPC education. According to Cisco Visual Networking Index [4], nearly 79% of the world's mobile data traffic will be video by 2022. In order to maximize the gains of video on HPC education platform, it is imperative for video content providers to ensure good quality of experience (QoE).

Generally, a video can be encoded in constant bitrate (CBR) and variable bitrate (VBR) modes. CBR mode encodes the chunks on the same track with a relatively fixed bitrate. In contrast, VBR mode allocates higher bitrates to the complex scenes while maintaining an average bitrate throughout. Recently, video content providers have moved towards adopting VBR encoding since it presents the ability to realize better video quality with the same average bitrate than CBR [13].

It is exceptionally difficult to provision and share video contents without the right set of streaming methodology. Adaptive bitrate (ABR) algorithms have emerged as the primary tool to improve QoE. The video streaming servers encode original videos into multiple tracks with different bitrate levels and segment the videos into equal-duration (e.g., 4s) chunks. ABR algorithms run at the client-side and adaptively select a suitable bitrate for each video chunk underlying a dynamic network condition. It is challenging to design an efficient ABR algorithm due to the conflicting goals of QoE metrics (high quality, minimal rebuffering, smoothness, etc.) and inherent instability of network throughput. Prior works developed ABR schemes based on different factors such as estimated network throughput and playback buffer occupancy via fixed rules, control theoretic-based methods, deep learning-based methods, etc.

Despite the above works, challenges remain in the scenario of VBR video streaming: 1) The chunk sizes of VBR videos vary rapidly, and the risk of rebuffering increases when streaming complex chunks. 2) The cascading effects of bitrate selection is larger compared to CBR mode (e.g., selecting a high bitrate level for a complex chunk consumes much bandwidth and buffer). 3) the quality of complex chunks suffers degradation in a low bitrate level, which leads to poor QoE metrics. Most existing ABR schemes designed for CBR encoding are not suitable to stream VBR videos because they typically assume a track's average bitrate to be the bandwidth requirement.

To address the above challenges, we propose VBSSR (Variable Bitrate encoded video Streaming with Super-Resolution), a novel ABR scheme for VBR video streaming. VBSSR adopts the technique of super-resolution (SR) to boost the quality of video chunks at the client-side, trading the client's computation capacity for a reduction of bandwidth consumption. It is intuitive that streaming complex chunks with relatively low bitrate levels can save more bandwidth compared to simple chunks. Moreover, the profit of enhancing complex chunks is higher than that of simple ones. In particular, VBSSR selectively enhances a part of chunks (e.g., complex chunks) rather than all of them to meet the real-time requirement. A deep reinforcement learning (DRL)-based model is designed to adaptively choose the bitrate level for each chunk and make the decision on which chunks to be enhanced.

In summary, our contributions in this paper can be concluded as below:

- We propose a new framework called VBSSR for ABR streaming in views of VBR encoding videos, making full use of the computation capacity of clients to improve QoE and save numerous network bandwidth.
- We carefully design a deep reinforcement learning algorithm to select the bitrate level for each chunk and decide which chunks to be enhanced by SR to boost the video quality.
- We evaluate the performance of our proposed method through extensive experiments. The simulation results show that VBSSR outperforms the existing ABR scheme with significant improvement of QoE metrics, while effectively reducing the rebuffering time and bandwidth consumption.

The rest of the paper is structured as follows. In Sect. 2, we first review the existing related works. Then, we introduce the system model and formally formulate the problem in Sect. 3. The details of our DRL-based ABR algorithm is proposed in Sect. 4. In Sect. 5, we conduct a series of experiments to provide performance evaluation. We conclude this work in Sect. 6.

2 Related Work

With the explosive growth of HTTP-based video traffic, the research on the ABR algorithm has become pretty active in recent years. The existing works consider different influence factors to study bitrate adaption for optimizing QoE.

Rate-based (RB) methods [8, 17] estimate the bandwidth according to the past observed network throughput and make choices based on the estimations. In contrast, buffer-based (BB) [5, 7, 16] methods pay attention to the client’s playback buffer occupancy. As the most recent buffer-based approach, BOLA [16] is an online control algorithm using Lyapunov optimization and does not require the prediction of available network bandwidth. Furthermore, *Yin et al.* [22] combine the above two elements and maximize QoE over a horizon of several future chunks with Model Predictive Control (MPC). PIA [14] is another control-theoretic ABR framework leveraging Proportional-Integral-Derivative (PID) control concepts to maintain a target buffer level and it occurs smaller runtime overhead compared to MPC. Oboe [1] integrates several basic ABR schemes and adaptively switches schemes to select bitrates. The more intelligent existing works introduce machine learning to “learn” ABR schemes without any presumptions. *Mao et al.* [10] proposed Pensieve that first applies DRL algorithms to train RL agents based on the network throughput traces and video data. *Huang et al.* [6] presented a video quality-aware ABR approach called Comyco to train policy via imitation learning.

However, the above schemes are designed for the CBR encoded videos and they simply use average bitrates to make adaption decisions. Therefore, most of them suffer QoE degradation when streaming VBR encoded videos whose chunk sizes are violently fluctuant even in the same bitrate level. There are few works focus on VBR video streaming in recent years. *Qin et al.* [13] analyzed the

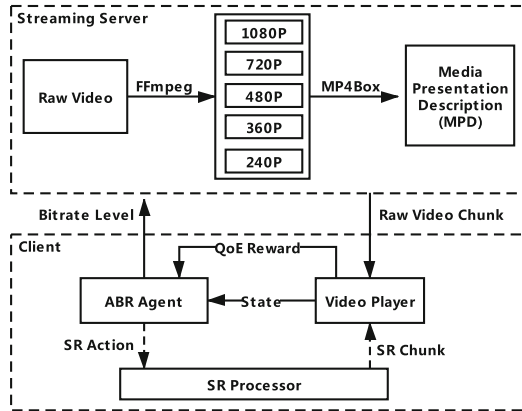


Fig. 1. System overview

distinguishing characteristics of VBR encoding and presented CAVA using PID control-theoretic to deliver high-quality chunks. But this scheme is relatively conservative. Benefit from the development of the client’s computation capacity, NAS [21] is proposed to boost the quality of video chunks by adopting super-resolution based on deep neural networks (DNNs). Different from the prior works, we take the characteristics of VBR encoding into consideration and integrate SR technique into VBR encoded video streaming to optimize QoE metrics.

3 System Model and Problem Formulation

In this session, we first introduce the framework of our proposed VBSSR system, then provide the problem formulation and our goal.

3.1 System Overview

In the VBSSR system, the super-resolution operations are conducted at the client-side to enhance the quality of low-resolution chunks and reduce the bandwidth cost. Figure 1 illustrates the system model, which consists of the following components.

Streaming Server: The streaming server is responsible for preprocessing the raw videos and delivering the video content to the client. In the preprocessing stage, each video is encoded into multiple resolution versions at different bitrate levels with FFmpeg [3] and split into equal-duration chunks. Then, the server uses MP4Box to generate the Media Presentation Description (MPD) manifest file that involves the information of each chunk. Since we focus on Video-on-Demand (VoD) service in this paper, the video encoding and MPD generation can be done before video streaming.

Client: The client mainly includes a video player, an ABR agent and an SR processor. At the beginning of the video play, the client fetches the video chunks

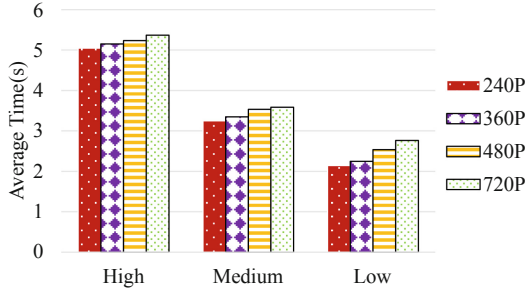


Fig. 2. Average SR inference time for 4-second chunk using Nvidia Gxforce 1080

and SR model parallelly. In this stage, the client downloads video chunks with the lowest bitrate level to avoid rebuffering. When the SR model is fully downloaded, the ABR agent begins to perform actions, including the bitrate selection for each chunk and the decision whether to enhance the chunk quality by SR. Once taking an action, the client sends a request to the server and downloads the specified chunk. If the action contains SR operation, the SR processor will enhance the chunk and replaces the original chunk in the playback buffer when the enhancement process is completed.

3.2 Content-Aware Super-Resolution

The content-aware super-resolution utilizes computation resources of clients to boost the video quality. It can learn the features that map the low-resolution video to the high-resolution video. However, it is challenging to meet the real-time requirement, since the inference time of SR is proportional to the video quality enhancement and the available computation capacity changes across clients. To address this problem, the work [21] provides multiple levels of DNNs that vary in quality and computational requirements. We train three SR models (e.g., ‘Low’, ‘Medium’ and ‘High’) with the same configurations as [21] on desktop PC using an Nvidia GTX 1080 GPU, and record their average inference time for each 4-second chunk. The result is shown in Fig. 2.

As shown in Fig. 2, the system can at most use the ‘Medium’ model to satisfy the real-time requirement. Nevertheless, for the VBR encoded videos, the quality improvement of complex chunks with the ‘High’ model is much higher than the ‘Medium’ one, while the enhancement of simple chunks is inapparent. Therefore, it is beneficial for the client to transmit complex chunks with low bitrate levels and enhance them by SR. When steaming the simple chunks, it can choose high bitrate levels without SR operation. In this case, the system needs to determine which chunks to be enhanced.

To address the above challenge, we design an SR model by extending the scalable neural network in [21], which supports multi-scale input (x_1 , x_2 , x_3 , x_4). The SR model maintains a queue that stores the chunks to be enhanced. Whenever the SR operator finishes processing a chunk and the original video chunk

has not been played, it will replace the original chunk in the buffer. Otherwise, the incomplete chunk in the queue will be abandoned.

3.3 Problem Formulation

In this paper, the streaming server encodes a video into m bitrate levels and segments them into n equal-duration chunks, each of which contains l seconds. The maximum player buffer size of the client is indexed as B .

Let $a_i = \{a_i^1, a_i^2\}$ be the action that ABR agent makes for c_i , where $a_i^1 \in \{1, 2, \dots, m\}$ denotes the bitrate level to download, and $a_i^2 \in \{0, 1\}$ determines whether to perform SR operation on c_i . We define the current player buffer size as B_i , where $B_i \in [0, B]$, and the length of SR queue as Q_i . The download time of the c_i is indexed as D_i , then the buffer size for next chunk can be formulated as:

$$B_{i+1} = \begin{cases} (B_i - D_i)^+ + l - \Delta t, & \text{if } (B_i - D_i)^+ + l > B \\ (B_i - D_i)^+ + l, & \text{otherwise.} \end{cases} \quad (1)$$

where $(X)^+ = \max(X, 0)$. If $B_i - D_i < 0$, the rebuffer event will occur. If the buffer size is larger than B , the client will stop fetching chunks and wait for a fixed duration Δt . For simplicity, we set $\Delta t = l$.

If $a_i^2 = 1$, the chunk will be put into the SR queue. The inference time of this chunk is denoted as T_i , the result whether the SR operation satisfies the real-time requirement can be defined as:

$$q_i = \begin{cases} 1, & \text{if } B_i - Q_i - T_i \geq 0, \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

if $q_i = 0$, the client will play the original chunk. Note that if $a_i^2 = 0$, $q_i = 0$ as well.

Similarly, the SR queue dynamics can be formulated as:

$$Q_{i+1} = \begin{cases} (Q_i - D_i)^+ + T_i, & \text{if } q_i = 1, \\ (Q_i - D_i)^+, & \text{otherwise.} \end{cases} \quad (3)$$

In video streaming, the key elements of QoE metrics can be given as follows:

$$QoE = \underbrace{\sum_{i=1}^n R_i}_{\text{quality}} - \mu \underbrace{\sum_{i=1}^n (D_i - B_i)^+}_{\text{video stall}} - \omega \underbrace{\sum_{i=1}^{n-1} |R_{i+1} - R_i|}_{\text{quality variation}} \quad (4)$$

where μ and ω are the penalty weights of rebuffering and quality changes. R_i is the bitrate of c_i . If $q_i = 1$, to reflect the SR-based quality enhancement, R_i will be calculated as follows:

$$\tilde{R}_i = SSIM^{-1}(SSIM(DNN(R_i))) \quad (5)$$

where $DNN(R_i)$ represents the quality enhanced by the SR model. SSIM is the average structural similarity to measure the chunk quality [19], and its reverse $SSIM^{-1}$ maps an SSIM value back to the bitrate. In this paper, VBSSR aims to maximize QoE metrics (4).

4 DRL-Based Algorithm Design

In the above problem, we need to optimize multiple objectives and make a series of choices. In this section, a DRL-based ABR algorithm is proposed to solve the optimization problem. We first specialize the key elements of our DRL model, then introduce the detail of the training algorithm.

4.1 Key Elements of DRL Model

There are three key elements in our DRL-based ABR model, i.e., state, action, and reward.

- **State:** The state is the input of the DRL neural networks. After the download of each chunk i , the state s_i can be represented by:

$$s_i = \left(R_i, B_i, C_i, Q_i, \vec{X}_i, \vec{D}_i, \vec{T}_i, \vec{S}_i \right) \quad (6)$$

where R_i is the bitrate chosen for the last chunk; B_i is the current buffer occupancy; Q_i is the current SR queue length; C_i is the number of remaining chunks till the end of the video; \vec{X}_i , \vec{D}_i and \vec{T}_i represent the average network throughput, download time and SR inference time for the past p chunks, respectively; \vec{S}_i is the vector of m available sizes for the next chunk.

- **Action:** The agent takes an action a_i for each $chunk_i$ to decide the bitrate level and SR operation. Its definition has been declared in Sect. 3.3.
- **Reward:** The reward of the i th chunk is defined as follows:

$$r_i = R_i - \mu(D_i - B_i)^+ - \gamma |R_i - R_{i-1}| - \sigma P_i \quad (7)$$

$$P_i = \begin{cases} \tilde{R}_i - R_i, & \text{if } a_i^2 = 1 \text{ and } q_i = 0, \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

P_i is the penalty term of the failed SR operation and σ is the weight. If $a_i^2 = 1$ but c_i is failed to be enhanced before being played, the agent will receive the punishment. We add this term for better making the SR choice.

4.2 Policy and Training Algorithm

The agent makes its action based on a policy $\pi : (s_i, a_i) \rightarrow [0, 1]$, which means the probability distribution of the action a_i being taken in the state s_i . To overcome the challenge of high-dimensional state and action space, we use a neural network with parameters θ to obtain the policy $\pi_\theta : (s_i, a_i)$. In particular,

we adopt the state-of-the-art actor-critic framework A3C [11] to train the neural network where the gradient can be computed by:

$$\nabla_{\theta} \mathbb{E}_{\pi_{\theta}} \left[\sum_{i=1}^n \gamma^i r_i \right] = \mathbb{E}_{\pi_{\theta}} [\nabla_{\theta} \log \pi_{\theta}(s, a) A^{\pi_{\theta}}(s, a)] \quad (9)$$

where γ is the discount factor and $A^{\pi_{\theta}}(s, a)$ is the advantage function indicating how better the action a is better than other actions following the policy in state s . In practice, the agent samples a trajectory of selected bitrate levels and SR operations, then empirically estimates an unbiased $A^{\pi_{\theta}}(s_i, a_i)$ as $A(s_i, a_i)$. The parameters of the actor-network can be updated as below:

$$\theta = \theta + \alpha_1 \sum_i \nabla_{\theta} \log \pi_{\theta}(s_i, a_i) A(s_i, a_i) \quad (10)$$

where α_1 is the learning rate of the actor network.

The critic-network can be updated as follows:

$$\theta_v = \theta_v - \alpha_2 \sum_i (r_i + \gamma V^{\pi_{\theta}}(s_{i+1}; \theta_v) - V^{\pi_{\theta}}(s_i; \theta_v))^2 \quad (11)$$

where α_2 is the learning rate, and $V^{\pi_{\theta}}(\cdot; \theta_v)$ is the output of the critic network.

5 Performance Evaluation

5.1 Experiment Settings

DRL Model Settings: For the actor-network, VBSSR input the last chunk’s bitrate, the current buffer occupancy, the number of chunks left, and the length of current SR queue into four identical linear layers with 128 neurons. The $p = 8$ past chunk download delays and average network throughput measurements are passed to two identical 1D convolution layers (CNN) which contains 128 filters, each of size 4 with stride 1. Next chunk sizes are passed to another 1D-CNN with the same shape. The outputs from these layers are passed to a fully connected layer whose neurons are the same as the size of action space. The critic-network uses the same network structure but its output is only one neuron. The learning rates of actor and critic are set to 0.0001 and 0.001. We set the discount factor $\gamma = 0.99$, the penalty weight of failed SR operation $\sigma = 2$, and the maximum buffer size $B = 60$ s.

VBR Encoded Video Dataset: We download the video Elephant Dream from [20]. The video is encoded into 5 levels with multiple average bitrates (e.g., {400, 800, 1200, 2400, 4800}Kbps, with resolutions of {240, 360, 480, 720, 1080}p, respectively) using FFmpeg. To generate VBR videos, we set the maximum bitrates to twice the average bitrates. Each video is segmented into 4-second chunks by MP4Box. Totally, there are 140 chunks with an overall length of 560 s.

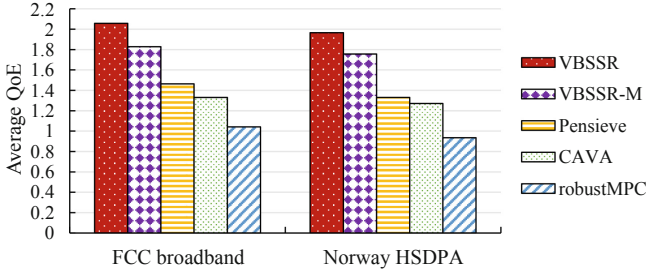


Fig. 3. Average QoE for each ABR scheme

Network Traces: The broadband dataset released by FCC [2] and HSDPA mobile dataset collected in Norway [15] are used to simulate network throughput. Each trace spans about 10 min. And we employ Mahimahi [12] to simulate the network condition from these traces.

Baselines: We compare the performance of VBSSR with four algorithms:

- **robustMPC:** uses Model Predict Control (MPC) to select a bitrate that maximizes the QoE over a horizon of 5 future chunks and accounts for the error between predicted and observed throughput.
- **Pensieve:** the first ABR scheme adopting deep reinforcement learning and achieve the state-of-the-art.
- **CAVA:** the first scheme in views of VBR video streaming, using PID control to maintain a target buffer level and handle the characteristic of VBR encoded video.
- **VBSSR-M:** a variation of VBSSR which enhances all chunks in the video with the ‘Medium’ level SR model to satisfy the real-time requirement.

QoE Metric: We use the linear QoE metric, which is used for pensieve [10] and robustMPC [22], to evaluate the above schemes. In particular, the rebuffering penalty weight μ is set to 4.3, and the smoothness penalty term ω is set to 1.

5.2 Performance Comparison

We first compare VBSSR with other methods by calculating the average QoE of each video chunk. Figure 3 illustrates the average QoE for each scheme on the datasets of FCC broadband and Norway HSDPA. The results demonstrate that VBSSR outperforms the best of existing algorithms. In particular, the average QoE achieved by VBSSR is 37.1% and 47.7% higher than Pensieve on the FCC broadband dataset and Norway HSDPA dataset, respectively. VBSSR outperforms CAVA with the improvements of 50.9% (for the FCC dataset) and 54.7% (for the Norway dataset). Moreover, although VBSSR-M enhances every chunk, VBSSR gains a better QoE compared to VBSSR-M because the improvement achieved by the ‘Ultra’ model is higher than the ‘Medium’ one. We also give the cumulative distribution function (CDF) of the average QoE in Figs. 4 and 5.

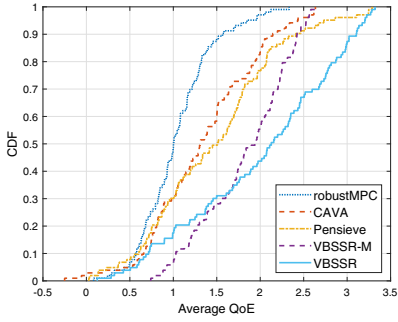


Fig. 4. Average QoE on the FCC broadband dataset

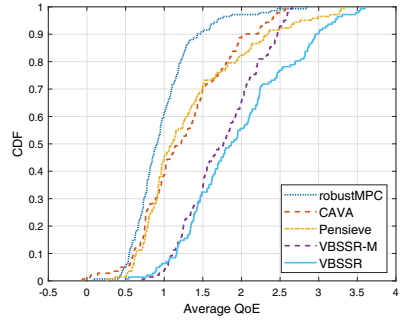


Fig. 5. Average QoE on the Norway HSDPA dataset

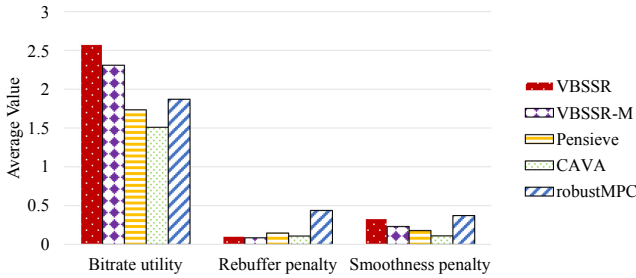


Fig. 6. Average values of the individual terms in the QoE metric on the FCC broadband dataset

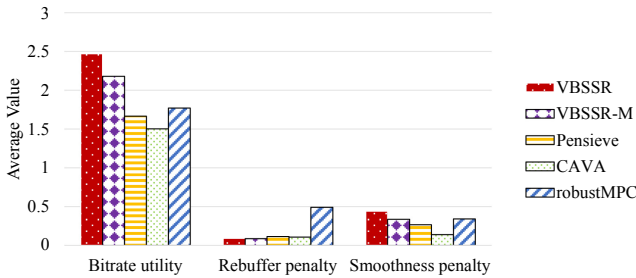


Fig. 7. Average values of the individual terms in the QoE metric on the Norway HSDPA dataset

To better understand QoE obtained by VBSSR, we analyze the performance on each individual term in the QoE objective. Figure 6 and 7 compare VBSSR with other algorithms in terms of average bitrate, rebuffering time and quality variations. As shown, VBSSR obtains the highest average bitrate which benefits from our rate adaptation and super-resolution enhancement. For the rebuffering

time, VBSSR is almost the same as VBSSR-M and lower than other methods. However, VBSSR suffers a little higher penalty of average quality variation. This is because the decision on SR enhancement may enlarge the quality change between the adjacent chunks. We note that robustMPC achieves a higher average bitrate than Pensieve and CAVA, but it suffers most rebuffering due to its relatively aggressive policy. In contrast, CAVA is designed for the VBR encoded video and it adopts a comparatively conservative strategy to maintain a target buffer level with a feedback control loop, thus it performs well on both rebuffering and quality variation. Pensieve is more intelligent and finds a better trade-off among QoE objectives. Overall, VBSSR optimizes the different terms of QoE compared with the rest ABR schemes.

6 Conclusion

In this paper, we build a novel ABR scheme called VBSSR to stream VBR encoded videos. VBSSR trends to fetch complex video chunks with relatively low bitrate levels to reduce network bandwidth consumption and boost the video quality at the client-side with SR. We train a DRL-based neural network to jointly choose suitable bitrates and decide which chunks to be enhanced. By conducting extensive trace-driven evaluations, we compare the performance of VBSSR with other state-of-the-art methods. The results show that our method significantly outperforms other approaches with higher quality and lower rebuffering penalty, achieving the best QoE metrics overall.

Acknowledgement. This work was supported by the National Key R&D Program of China under Grant 2018YFB0204100, the National Natural Science Foundation of China under Grants U1911201, 61802452, 62072486, Guangdong Special Support Program under Grant 2017TX04X148, and the project “PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019)”


References

1. Akhtar, Z., et al.: Oboe: auto-tuning video ABR algorithms to network conditions. In: Proceedings of the Conference of the ACM Special Interest Group on Data Communication, pp. 44–58 (2018)
2. Commission, F.C.: Raw data - measuring broadband America (2016). <https://www.fcc.gov/reports-research/reports/>
3. Ffmpeg: Ffmpeg project (2017). <https://www.ffmpeg.org/>
4. Forecast, G.: Cisco visual networking index: Global mobile data trafficforecast update 2017–2022. Update (2019)
5. Huang, T.Y., Johari, R., McKeown, N., Trunnell, M., Watson, M.: A buffer-based approach to rate adaptation: evidence from a large video streaming service. In: Proceedings of the 2014 ACM Conference on SIGCOMM, pp. 187–198 (2014)
6. Huang, T., Zhou, C., Zhang, R.X., Wu, C., Yao, X., Sun, L.: Comyco: quality-aware adaptive video streaming via imitation learning. In: Proceedings of the 27th ACM International Conference on Multimedia (MM), pp. 429–437. Association for Computing Machinery, New York (2019)

7. Huang, W., Zhou, Y., Xie, X., Wu, D., Chen, M., Ngai, E.: Buffer state is enough: simplifying the design of QoE-aware HTTP adaptive video streaming. *IEEE Trans. Broadcast.* **64**(2), 590–601 (2018)
8. Jiang, J., Sekar, V., Zhang, H.: Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In: *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*, pp. 97–108 (2012)
9. Luo, Z., Wang, Z., Wu, D., Hei, X., Du, Y.: Automatic generation and assessment of student assignments for parallel programming learning. In: Shen, H., Sang, Y. (eds.) *PAAP 2019. CCIS*, vol. 1163, pp. 184–194. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2767-8_18
10. Mao, H., Netravali, R., Alizadeh, M.: Neural adaptive video streaming with pen-sieve. In: *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pp. 197–210 (2017)
11. Mnih, V., et al.: Asynchronous methods for deep reinforcement learning. In: *International Conference on Machine Learning*, pp. 1928–1937 (2016)
12. Netravali, R., et al.: Mahimahi: accurate record-and-replay for HTTP. In: Lu, S., Riedel, E. (eds.) *Proceedings of USENIX Annual Technical Conference (ATC)*, pp. 417–429. USENIX Association (2015)
13. Qin, Y., et al.: ABR streaming of VBR-encoded videos: characterization, challenges, and solutions. In: *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*, pp. 366–378 (2018)
14. Qin, Y., et al.: A control theoretic approach to ABR video streaming: a fresh look at PID-based rate adaptation. *IEEE Trans. Mob. Comput.* **19**(11), 2505–2519 (2019)
15. Riiser, H., Vigmostad, P., Griwodz, C., Halvorsen, P.: Commute path bandwidth traces from 3G networks: analysis and applications. In: *Proceedings of the 4th ACM Multimedia Systems Conference*, pp. 114–118 (2013)
16. Spiteri, K., Urgaonkar, R., Sitaraman, R.K.: BOLA: near-optimal bitrate adaptation for online videos. *IEEE/ACM Trans. Networking* **28**(4), 1698–1711 (2020)
17. Sun, Y., et al.: CS2P: improving video bitrate selection and adaptation with data-driven throughput prediction. In: *Proceedings of the ACM SIGCOMM*, pp. 272–285 (2016)
18. Wang, Z., Wu, D., Luo, Z., Du, Y.: Building a lightweight container-based experimental platform for HPC education. In: Shen, H., Sang, Y. (eds.) *PAAP 2019. CCIS*, vol. 1163, pp. 175–183. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2767-8_17
19. 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)
20. Xiph.Org: Video test media (2016). <https://media.xiph.org/video/derf/>
21. Yeo, H., Jung, Y., Kim, J., Shin, J., Han, D.: Neural adaptive content-aware internet video delivery. In: *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 645–661 (2018)
22. Yin, X., Jindal, A., Sekar, V., Sinopoli, B.: A control-theoretic approach for dynamic adaptive video streaming over HTTP. In: *Proceedings of the ACM Conference on Special Interest Group on Data Communication*, pp. 325–338 (2015)



An Investigation on the Performance of Highly Congested Home WiFi Networks During the COVID-19 Pandemic

Rhys Cunningham, Ben Hendry, and Wee Lum Tan^(✉) 

School of ICT, Griffith University, Brisbane, Australia
{rhys.cunningham, ben.hendry}@griffithuni.edu.au,
w.tan@griffith.edu.au

Abstract. Due to COVID-19, many people throughout the world have had to suddenly transition to working and studying from home. This has led to an increase in data traffic in the home WiFi networks using the 802.11ac or 802.11n routers. In this paper, we have investigated the performance of the home WiFi networks in supporting the quality-of-service (QoS) needs of multiple home users concurrently watching streaming videos, playing online games, participating in online meetings, downloading huge files, etc. We measured the performance of these traffic streams using metrics such as delay, throughput, MOS-LQO (Mean Opinion Score - Listening Quality Objective) metric for audio streams and the VMAF (Video Multimethod Assessment Fusion) metric for video streams. Our results indicated that under highly congested network conditions, the 802.11ac router's default settings performs the best in supporting the multiple concurrent traffic streams. In particular, our results show that the WMM 802.11e QoS feature in the router is critical to the router's ability to prioritize video and audio traffic over other normal data traffic, and hence should always be enabled. Our results also show that the 802.11n router should be configured to operate in the 5 GHz frequency band. This is to avoid the congested 2.4 GHz frequency band and therefore enable the 802.11n router to support the QoS needs of the home users.

Keywords: IEEE802.11 performance evaluation · Quality of service · Throughput

1 Introduction

In 2020, the COVID-19 pandemic forced millions of employees and students to conduct their work and studies remotely. In May 2020, the Australian Bureau of Statistics conducted a survey that revealed the COVID-19 pandemic forced 46% of Australians to work and study from home [1]. From February to early April, the Australian National Broadband Network (NBN) reported an increase in data demand by “70 to 80 per cent during daytime hours” [2]. The sudden increase in data demand can be attributed to more home users accessing online content and applications, e.g. Zoom or Teams meetings, video streaming, online game playing, file download/upload, etc. Much interest has focused on the network capacity and speed of the Internet Service

Provider (ISP) infrastructure in supporting the Internet needs of home users. However, there are other factors that influence the quality-of-service (QoS) that a home user experiences, namely the network load on the servers (e.g. popular websites or streaming servers) and the performance of private residential WiFi infrastructure under these high-congestion conditions are also as important. In this paper, we focus on the performance of IEEE 802.11ac and IEEE 802.11n wireless links in highly congested home WiFi networks, and investigate the router settings that would optimize the performance of these home WiFi networks.

In this paper, we have investigated the network performance in two separate home network configurations: multiple 802.11ac-enabled devices connected to an 802.11ac router, and multiple 802.11n-enabled devices connected to an 802.11n router. The devices connected to the router in each case vary from handheld mobile devices to laptops, reflecting the heterogeneous nature of typical home networks. These multiple devices concurrently perform one of the following activities: bulk data transfer, video streaming, video gaming, and audio conferencing, with these concurrent data streams replicating the high-congestion conditions typically encountered in a home network under work-from-home scenarios. Network performance is evaluated using metrics such as delay, throughput, PESQ MOS-LQO [3], and VMAF [4]. We carried out several tests and measured the network performance under different router settings: default, shorter guard interval, 802.11e QoS disabled, higher channel bandwidth, and different frequency band. Our 802.11n results indicated that network performance is the best when the 802.11n router is configured to operate in the 5 GHz frequency band. The 802.11ac results showed that the 802.11ac router default settings produce the best network performance and that the 802.11e QoS feature is critical to the ability of the router to prioritize different traffic streams and provide sufficient QoS to the home users under highly congested network conditions.

The rest of the paper is structured as follows. In Sect. 2, we discuss some related work while Sect. 3 describes our test setup and methodology. Experimental results are presented in Sect. 4, and we provide our conclusions in Sect. 5.

2 Related Work

Since the release of IEEE 802.11n and IEEE 802.11ac, researchers have conducted numerous studies to evaluate and improve the performance of these networks. The 802.11n standard defined a theoretical data transmission rate of up to 600 Mbps, using four spatial streams in a 40 MHz channel with a 400 ns guard interval [5], with a 200 Mbps throughput more typically achieved in a conventional, non-laboratory setting [6]. Conversely, the 802.11ac Wave 1 standard defined a theoretical data transmission rate of up to 1300 Mbps, using three spatial streams, with a 400–700 Mbps throughput more typically achieved in a conventional, non-laboratory setting [6].

As the successor, 802.11ac significantly outperforms 802.11n in an indoor environment, with data rates exceeding 700 Mbps for a 3×3 multiple-input and multiple-output (MIMO) configuration [7]. However, suboptimal channel conditions, such as distance, physical obstacles, and interference in the 5 GHz frequency band from other wireless technologies, decrease the magnitude of these performance improvements

markedly. Similarly, [8] noted that 802.11n provided the worst average throughput due to the highly congested 2.4 GHz band in their experimental environment.

[9] discovered 802.11ac performed adequately in the transmission of compressed 4kUHD (Ultra High Definition) with the transmission range not exceeding 10 m. [10] found 802.11n supported some low-rate voice/video connections but remained limited in its capacity to transmit high-definition video; though, [11] noted the quality of video streaming transmitted at 5 GHz outperformed the 2.4 GHz transmitted signal. [12] attributed the increased throughput achieved by 802.11ac to the following mechanisms: multi-user multiple-input multiple-output, larger channel bandwidths of 80 and 160 MHz, and 256-quadrature amplitude modulation (QAM).

The selection of codecs greatly impacts the quality-of-experience (QoE) of Voice over IP (VoIP) for 802.11 wireless network users. For 802.11ac networks, [13] noted the G.729 VoIP codec achieved the highest throughput, while the G.723 VoIP codec achieved the lowest throughput of their tested codecs.

The utilisation of larger channel bandwidths is less energy efficient due to its higher power consumption in idle listening mode [14]. However, increasing the bandwidth improves throughput significantly, although unplanned selection of primary channels and channel bandwidths may severely degrade the throughput of links operating at larger channel bandwidths. [7] found co-channel interference greatly reduced the throughput in 802.11ac WLANs. Likewise, [15] noted 802.11ac networks are quite sensitive to other transmissions on overlapped channels, which caused a significant reduction in throughput. However, 802.11ac ensures reliability through its ability to sub-divide transmissions in larger 40 MHz or 80 MHz channels into 20 MHz channels. Such a mechanism ensures that while throughput may be reduced in 802.11ac networks, the networks do not experience breaks in communication.

3 Experimental Methodology

In this section, we describe the performance metrics, hardware and software tools used in the 802.11ac and 802.11n tests and the experimental methodology. All our tests are carried out in a home setting, in the early hours of the morning in order to minimize potential interferences from other home networks.

Table 1 shows the hardware used in our tests, while the software tools used in our tests are:

- iPerf: Used for generating TCP traffic in the bulk data transfer tests.
- Open Broadcaster Software (OBS): Used to send, receive, and record video traffic.
- Team Fortress 2: Used to generate video game traffic.
- Zoom: Plays an audio file to generate VoIP traffic.
- Wireshark: Used to capture sent and received packets for analysis.
- VMAF Development Kit (VDK): Used to compare and analyze the sent and received video files.
- PESQ (P.862): Used to compare and analyze the sent and received audio files.
- Audacity: Used to record and prepare received audio from the conferencing environment for comparison with the PESQ software.

Table 1. Hardware Used in 802.11ac and 802.11n Tests.

Device	Model	Purpose
A	Samsung Galaxy S9+	Bulk Data Transfer
B	Alienware 14	Video Streaming
C	Dell Inspiron 3580	Video Gaming
D	Dell Inspiron 7570	Audio Conferencing
E	Asus Desktop	Server
802.11ac Router	Huawei HG659	Router
802.11n Router	Netgear N600	Router

The metrics used to measure network performance in our tests are:

- Delay: Time taken for a packet to travel from the sender to the receiver, measured in milliseconds (ms).
- Throughput: The amount of data transferred per second from the sender to the receiver, measured in Mbps.
- MOS-LQO: The QoE metric for audio. MOS-LQO ranges between 1.02 (worst) and 4.56 (best) [16].
- VMAF score: The QoE metric for video. The VMAF score ranges between 0 (worst) and 100 (best) [4].

3.1 802.11ac Tests

We first performed a baseline measurement test to determine the maximum saturation throughput achievable for a single device in an 802.11ac wireless link. The testbed is shown in Fig. 1 where the desktop server is connected via a 1 Gbps Ethernet cable to the 802.11ac router. Using iPerf, the desktop server sent TCP traffic downlink (at the maximum permissible sending rate) to the laptop, and the maximum saturation throughput is measured at the laptop. Table 2 shows the configuration settings of the router in the baseline test.

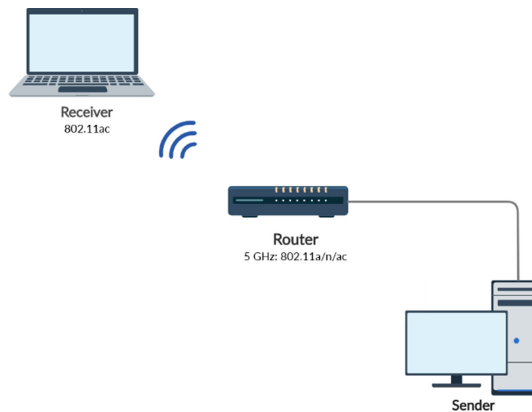


Fig. 1. The 802.11ac network topology to measure the baseline network performance

Table 2. 802.11ac Default Router Settings in Baseline Measurement Test.

Setting	Value
Transmission Mode	802.11a/n/ac
Channel	Auto
WMM (802.11e QoS)	Enabled
Modulation & Coding Scheme (MCS)	Auto
Bandwidth	20/40 MHz (Auto)
Guard Interval	Long

We repeated the baseline measurement test for the following traffic streams: video streaming, audio conferencing, and video gaming, with the desktop server configured to be a video streaming server, an audio conferencing peer, and a video game server.

After the baseline measurement tests, we performed the high-congestion measurement tests where multiple, concurrent traffic streams (bulk TCP data transfer, video streaming, audio conferencing, and video gaming) on multiple end-devices are competing for bandwidth from the single 802.11ac router, as shown in Fig. 2 below. The high-congestion measurement tests were first conducted for the 802.11ac default router settings shown in Table 2. Next, we change a single router setting (one at a time) to determine its impact on the network performance, as follows:

- WMM (802.11e QoS) is disabled – this setting prioritises packets for voice and video services. Without WMM QoS, packets are treated with equal priority.
- Short guard interval – reduced to short (400 ns) from long (800 ns). A short guard interval potentially increases throughput but is more susceptible to interference.
- 80 MHz channel bandwidth – increased from the default 20/40 MHz (auto).

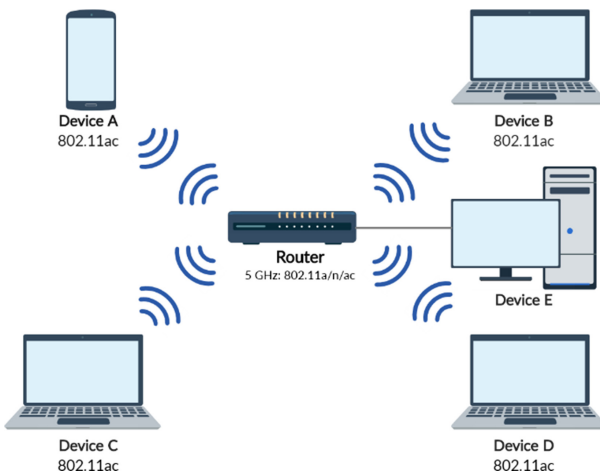


Fig. 2. The 802.11ac network topology to measure performance and QoS in a high-congestion home network with multiple, concurrent traffic streams

Table 3. 802.11n Default router settings in baseline measurement test

Setting	Value
Transmission Mode	802.11b/g/n
Channel	Auto
Bandwidth	20 MHz
Frequency Band	2.4 GHz

3.2 802.11n Tests

Similar to the 802.11ac tests, the 802.11n testing methodology consisted of a baseline measurement test and a high-congestion measurement test. The network topology for the baseline measurement test is similar to that shown in Fig. 1, while the network topology for the high-congestion measurement test is similar to that shown in Fig. 2.

The 802.11n router in the baseline measurement test has the default settings shown in Table 3. We first measured the maximum saturation throughput achievable for a single device in an 802.11n wireless link. We repeated the baseline measurement test for the other traffic streams (video, audio, game) with the default router settings in Table 3.

We next carried out the high-congestion measurement tests where multiple concurrent traffic streams (bulk TCP data transfer, video streaming, audio conferencing, and video gaming) on multiple end-devices are competing for bandwidth from the 802.11n router. The high-congestion measurement tests were first conducted for the default 802.11n router settings shown in Table 3. We then repeated the tests with the following settings to determine the impact on the network performance:

- Frequency band 2.4 GHz, channel bandwidth 40 MHz
- Frequency band 5 GHz, channel bandwidth 40 MHz

4 Results and Analysis

4.1 Baseline Test: Saturation Throughput

Figure 3 shows the results of the baseline test to measure the saturation throughput. In the 802.11ac baseline tests, a router with 20/40 MHz channel bandwidth achieved a saturation throughput of 142 Mbps for TCP traffic. By increasing the router's channel bandwidth to 80 MHz, the maximum TCP throughput increased to 270 Mbps.

In the 802.11n baseline tests, a router operating on the 2.4 GHz band with a 20 MHz channel bandwidth achieved a saturation throughput of 41 Mbps. Changing the channel bandwidth to 40 MHz increased the throughput slightly to 52 Mbps. On the 5 GHz band, the saturation throughput increased further to 111 Mbps.

These results indicate that by configuring the router to use a wider channel bandwidth and to operate on the 5 GHz frequency band (for the 802.11n router), we are able to obtain a higher throughput for the end users in the home WiFi network.

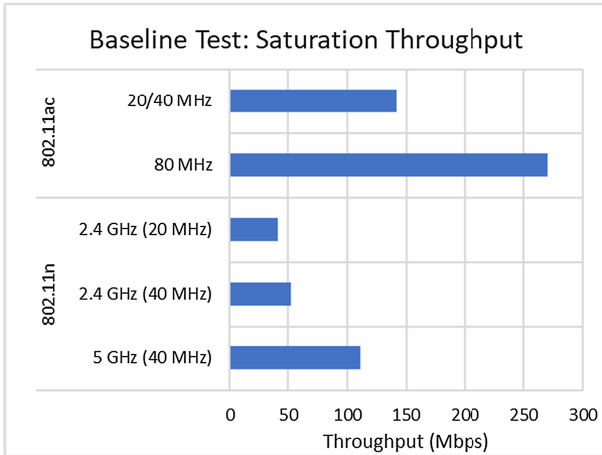


Fig. 3. The baseline 802.11ac and 802.11n saturation throughput performance

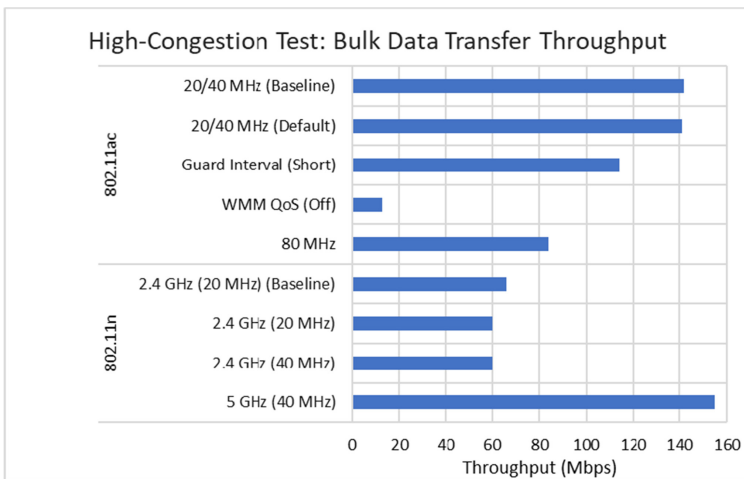


Fig. 4. The high-congestion 802.11ac and 802.11n bulk data transfer throughput performance

4.2 Bulk Data Transfer

Figure 4 to Fig. 7 show the results of the high-congestion test, where multiple concurrent traffic streams (bulk TCP data transfer, video streaming, audio conferencing, and video gaming) on multiple end-devices are competing for bandwidth from the 802.11ac router. We separately show the performance of each traffic stream, with Fig. 4 showing the throughput results of the bulk TCP data transfer.

In the 802.11ac test, we see that the throughput performance in the high-congestion scenario is similar to the throughput achieved in the baseline test, i.e. ~140 Mbps. This seemed to indicate that the 802.11ac router (with its default settings) is capable of

supporting the bulk data transfer traffic even when there are other competing traffic streams (e.g. video stream, audio stream, video game traffic) at the same time. However when the router setting is changed to scenarios with shorter guard interval, WMM QoS disabled, and wider channel bandwidth, we see that the throughput performance actually decreased. In particular when the WMM QoS feature is disabled, the achievable throughput decreased significantly to 13 Mbps. These results indicate that the 802.11ac router’s default settings produce the best throughput performance, and that the WMM 802.11e QoS feature in the router should never be disabled. The poorer throughput performance when the router is configured with shorter guard interval and wider channel bandwidth could be due to the interference from other networks during the test. In the 802.11n test, we see that the throughput performance is the best when the router is configured to operate in the 5 GHz frequency band with a channel bandwidth of 40 MHz.

4.3 Video Gaming

Figure 5 shows the delay performance of the video gaming traffic in the high-congestion test. For both the 802.11ac and 802.11n tests, we see that the delay performance is fairly similar across all scenarios (i.e. in the 20–30 ms range), with the exception of the scenario with WMM QoS disabled where the delay is a low 13.5 ms. This seemed counter-intuitive and we believe the delay performance in the WMM QoS disabled scenario may be due to the low number of sample results in our tests. Overall we see that the default settings in the 802.11ac router is sufficient to support the low delay requirements in video gaming. Similar to the bulk data transfer throughput results, the 802.11n router should be configured to operate in the 5 GHz frequency band to support video gaming delay requirements.

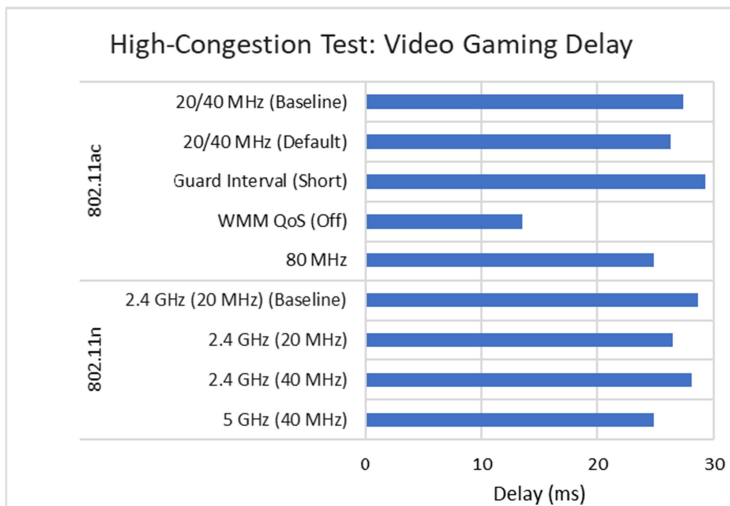


Fig. 5. The high-congestion 802.11ac and 802.11n video game delay performance

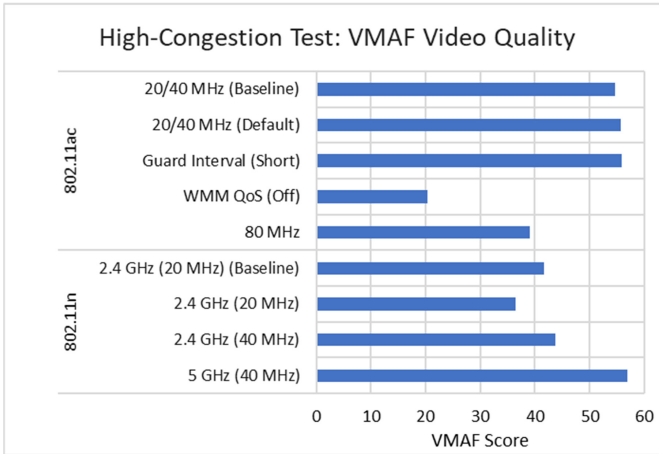


Fig. 6. The high-congestion 802.11ac and 802.11n VMAF score for video streaming

4.4 Video Streaming

For the video streaming traffic, the VMAF score served as the QoS metric for video quality. Figure 6 shows the VMAF comparison score of the source and received video streams. In the 802.11ac test, we see that the VMAF score in the high-congestion scenario is similar to the VMAF score achieved in the baseline test, i.e. ~ 55 . This seemed to indicate that the 802.11ac router (with its default settings) is capable of supporting the video streaming traffic even when there are other competing traffic streams (e.g. bulk data transfer traffic, audio stream, video game traffic) at the same time. We also see that the VMAF score is the best with the default settings in the router, and the worst VMAF score is achieved when the router is configured with the WMM QoS feature disabled. This result confirms the fact that the WMM QoS feature is critical to the ability of the router to prioritize video and audio traffic over other normal data traffic.

In the 802.11n test, we see that the VMAF score is higher when the router's channel bandwidth is increased to 40 MHz. However, this level of service is still incapable of streaming high-definition video at an acceptable level, especially in comparison to the QoS provided by the 802.11ac router. When the 802.11n router is configured to operate in the 5 GHz frequency band, the VMAF score is comparable to the scores achieved in the 802.11ac test. This again indicates that the 802.11n router should be configured to operate in the 5 GHz frequency band in order to properly support video streaming traffic.

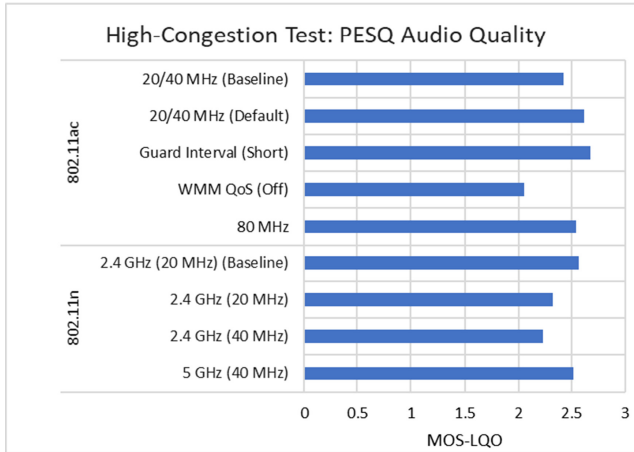


Fig. 7. The high-congestion 802.11ac and 802.11n MOS-LQO score for audio conferencing

4.5 Audio Conferencing

For audio conferencing traffic, the MOS-LQO value served as the QoS metric for audio quality. Figure 7 displays the MOS-LQO comparison values of the source and received audio files. Similar to the results seen in the video streaming test, we see that the best MOS-LQO value is achieved with the default settings in the 802.11ac router and the worst MOS-LQO value is seen when the WMM QoS feature is disabled in the router. This again confirms the importance of the WMM QoS feature in the router. For the 802.11n router, it again achieved the best MOS-LQO when it is configured to operate in the 5 GHz frequency band.

5 Conclusion

The COVID-19 pandemic has forced many people throughout the world to work and study from home. This has caused a data traffic increase in the home WiFi networks and the ISP's network infrastructure. In this paper, we are interested to know if the typical home WiFi networks are capable of supporting the quality-of-service (QoS) and Internet needs of users who are suddenly forced to work and study from home. We have investigated different settings of typical 802.11ac and 802.11n home routers and compare their network performance under highly congested traffic conditions. Our results indicate that the default settings on the 802.11ac routers are capable of supporting the multiple concurrent data streams consisting of bulk TCP data transfers, video streaming, audio conferencing and online video gaming traffic. In particular the WMM 802.11e QoS feature in the 802.11ac routers should always be enabled so as to allow the router to prioritize video and audio traffic over the other normal data traffic. Our results also indicate that the 802.11n routers should be configured to operate in the

5 GHz frequency band in order to support the QoS needs of multiple home users. This is because the default frequency band setting of 2.4 GHz is typically congested and used by other home networks.

In this paper, we have analyzed the performance of 802.11ac and 802.11n WiFi routers commonly utilized in home networks. However, the impending general adoption of the 802.11ax standard means an additional avenue of study is available. It would be interesting to also evaluate the performance of WiFi routers from other vendors. Ideally, this would include an 802.11ac router capable of utilizing a 160 MHz channel bandwidth.

References

1. ABS. (2020). 4940.0 - Household Impacts of COVID-19 Survey, 29 Apr - 4 May 2020.
2. Hobday, L., Sas, N.: Coronavirus affecting internet speeds, as COVID-19 puts pressure on the network. ABC News (2020)
3. https://en.wikipedia.org/wiki/Perceptual_Evaluation_of_Speech_Quality
4. <https://github.com/Netflix/vmaf>
5. Van Nee, R., Jones, V.K., Awater, G., Van Zelst, A., Gardner, J., Steele, G.: The 802.11 n MIMO-OFDM standard for wireless LAN and beyond. *Wirel. Pers. Commun.* **37**(3–4), 445–453 (2006)
6. Siddiqui, F., Zeadally, S., Salah, K.: Gigabit wireless networking with IEEE 802.11 ac: technical overview and challenges. *J. Netwk.* **10**(3), 164 (2015)
7. Dianu, M. D., Riihijärvi, J., Petrova, M.: Measurement-based study of the performance of IEEE 802.11 ac in an indoor environment. In 2014 IEEE International Conference on Communications (ICC), pp. 5771–5776. IEEE (2014)
8. Ravindranath, N.S., Singh, I., Prasad, A., Rao, V.S.: Performance evaluation of IEEE 802.11ac and 802.11n using ns3. *Ind. J. Sci. Technol.* **9**(26) (2016)
9. Adeyemi-Ejeye, A., Alreshoodi, M., Al-Jobouri, L., Fleury, M., Woods, J., Medhi, M.: IEEE 802.11 ac wireless delivery of 4kUHD video: The impact of packet loss. In: 2017 IEEE 7th International Conference on Consumer Electronics-Berlin, ICCE-Berlin, pp. 258–259. IEEE (2017)
10. Cai, L.X., Ling, X., Shen, X. S., Mark, J.W., Cai, L.: Supporting voice and video applications over IEEE 802.11 n WLANs. *Wirel. Netwk.* **15**(4), 443–454 (2009)
11. Ibrahim, N.K., Ali, A.H., Razak, M.R.A., Azhar, M.F.: The performance of video streaming over wireless-N. In 2012 IEEE Symposium on Wireless Technology and Applications (ISWTA), pp. 142–145 IEEE (2012)
12. Van Nee, R.: Breaking the gigabit-per-second barrier with 802.11 ac. *IEEE Wirel. Commun.* **18**(2), 4 (2011)
13. Malekzadeh, M.I.N.A., Ghani, A.A.A.: Hybrid LTE-802.11 AC network: QoS optimality evaluation of the voip codecs techniques. *J. Eng. Sci. Technol.* **14**(1), 279–290 (2019)
14. Zeng, Y., Pathak, P.H., Mohapatra, P.: A first look at 802.11 ac in action: Energy efficiency and interference characterization. In: 2014 IFIP Networking Conference, pp. 1–9 IEEE (2014)
15. Zankiewicz, A.: Experimental analysis of susceptibility of IEEE 802.11ac Wave 1 networks to adjacent and co-channel interference. *Przegląd Elektrotechniczny*, **94**(2), 88–91 (2018)
16. Zhao, Y., Yang, Y., Gao, Y.: Reproduction of MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis



Using Feed-Forward Network for Fast Arbitrary Style Transfer with Contextual Loss

Peixiang Chen, Yujie Zhang, Jiangyi Huang, and Zhanghui Liu^(✉)

College of Mathematics and Computer Science, Fuzhou University, Fujian, China
pxchen921@gmail.com, yjzhang021@gmail.com, wbyiml@gmail.com,
lzh@fzu.edu.cn

Abstract. Image style transfer is a method that extracts its style from the style image and applies this style to content image. Since the introduction of neural style transfer, the field of style transfer has developed rapidly, and many new methods have been proposed. There are also some methods based on feed-forward networks, but these methods can usually only transfer one style or several styles. And these methods usually use feature gram matrix to calculate style loss. However, due to the global nature of the gram matrix, some local details cannot be stylized well, which is prone to distortion and artifacts. In this work, we propose an arbitrary style transfer method based on feed-forward network, in which the gram matrix and feature similarity are used together to calculate the style loss. The loss calculated by the similarity between features (called contextual loss in this paper) is minimized to produce stylized images with better details and fewer artifacts. Experimental results and user study prove that our method has achieved the state-of-the-art performance compared with existing arbitrary style transfer methods.

Keywords: Arbitrary style transfer · Feed-forward network · Feature similarity

1 Introduction

The style transfer task takes a pair of content image and style image as input, and transfers the style in the style image to the content image. As shown in Fig. 1, the generated image is called a stylized image. In recent years, style transfer has been greatly developed with the emergence of neural style transfer based on convolutional neural networks. Gatys et al. [1] first proposed the use of convolutional neural networks to separate the content feature abstract representation and the style feature abstract representation of the image. High-level CNN features compactly encode semantic content, such as the object and global structure of the input image, while low-level CNN features encode local details, such as color and texture. The method of Gatys et al. [2] can transfer images of arbitrary styles, but its computational cost is very high and it takes a long time to generate a good image. In order to solve this problem, some methods based on feed-forward

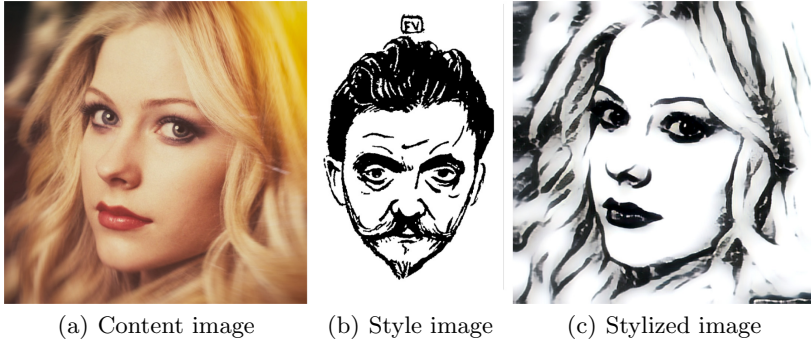


Fig. 1. An example of style transfer. (c) is the stylized image of our proposed method.

networks [6, 19] have been proposed. These methods can quickly generate stylized images. However, these methods have poor flexibility and cannot be migrated to arbitrary style images.

In order to achieve fast style transfer of arbitrary images, some new methods are proposed. The AdaIN method [4] proposes a novel adaptive instance normalization layer, which performs real-time arbitrary style transfer by matching the mean and variance of content features and style features. The WCT method [10] performs the whitening and coloring process through the pre-trained encoder-decoder. Li et al. [9] proposes a learnable linear transformation matrix for style transfer, which greatly reduces the computational cost and is much faster than the method of feature transformation [10, 11].

These mentioned methods all use the gram matrix of image features to represent the style of the image. However, we find that the gram matrix has limitations. The Gram matrix calculates the correlation between global features. Therefore, some local style feature information is lost, and a high-quality style image cannot be obtained. Inspired by the paper by Mechrez et al. [15], we introduce a loss function based on the similarity between features. Through the similarity between local features, we can obtain better stylized images that produce less distortion and artifacts.

In this work, our main contributions are as follows. First, we propose an arbitrary style transfer method based on a feed-forward network, which can perform fast style transfer and has better results than the state of the art arbitrary style transfer method. Second, we find the limitation of gram loss and introduce a loss function to be calculated by feature similarity. Combining this loss and gram loss, we propose a new style optimization objective that can effectively improve the quality of stylized images. Third, we find that for the style transfer task based on feed-forward network, too large batch size may lead to poor generalization ability. We explain this problem and conduct comparative experiments.

2 Related Work

2.1 Method Based on Image Optimization

Gatys et al. [2] iteratively update the white noise image by back-propagating content loss and style loss to obtain high-quality style transfer results. Risser et al. [17] introduced the histogram loss function to solve the problem of image texture disorder caused by unstable iterative optimization process. Yin et al. [21] proposed a content-aware style transfer algorithm, which can effectively control the synthesis of content images and style images, and further improve the quality of stylized images. Liao et al. [12] combined image analogy with deep learning and proposed a deep image analogy method that iteratively optimizes images through block matching. Due to the iterative optimization process, methods based on image optimization are computationally expensive, and these methods require several minutes or even tens of minutes to obtain a satisfactory result.

2.2 Method Based on Model Optimization

In order to solve the problem of low efficiency of iterative optimization, Johnson et al. [6] proposed a feed-forward method to achieve fast style transfer. This method uses the same content reconstruction loss and style reconstruction loss [2] to train a specific style generative model. This model can quickly generate stylized images of this style. Ulyanov et al. [19] proposed to replace batch normalization [5] with instance normalization [20] in the model training process, which significantly improved the quality of stylized images. Zhang et al. [24] constructed a generative model with multiple styles, which can quickly transfer multiple styles. Huang et al. [4] proposed an adaptive instance normalization method to achieve fast style transfer of arbitrary styles. Zhang et al. [23] proposed an arbitrary style transfer method based on meta-learning. For any style, only a few post-processing update steps are needed to quickly adapt to this style, which can achieve performance closed to the method [19].

2.3 Method Based on WCT (Whiten-Color Transform)

Li et al. [10] proposed to match the feature covariance of the content image with the feature covariance of the given style image. They regarded style transfer as an image reconstruction process, and mapped the statistical characteristics of the style image to the whitened content image. Li et al. [11] and Yoo et al. [22] developed WCT into photorealistic style transfer and achieved good results. Li et al. [9] theoretically derived the form of the transformation matrix, which was learned in a data-driven manner. Lu et al. [14] derived a closed-form solution by treating it as the optimal transport problem.

3 Proposed Method

Our model (Fig. 2) consists of a pair of pre-trained encoder-decoder, a pair of compress/decompress blocks, a residual module composed of multiple residual

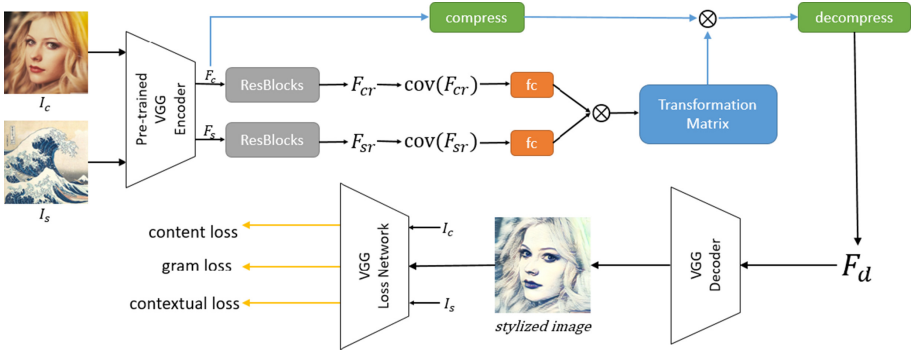


Fig. 2. Overview of the proposed model.

blocks, and a fully-connected layer. The VGG network [18] extracts features through the convolutional layer, and then the fully-connected layer classifies the image. The low-level features in the VGG network usually retain the color and texture information of the image. The high-level features lose parts of the color and texture information, but retain the semantic and structural information of the image. The pre-trained encoder is used to extract image features during the training process, and the decoder can faithfully restore the features to images. During the model training stage, this is fixed for the encoder and decoder. Because the high-level features extracted from the VGG encoder have high dimensions, we use a convolutional layer to reduce it to the size of the transformation matrix. This operation is called “compress”. Then we use a convolutional layer to decompress the transformed features to the corresponding dimensions. This operation is called “decompress”. In the training stage, the weight parameters of the pair of convolutional layers are learnable. Next we introduce in detail how the transformation matrix is trained.

3.1 Transformation Matrix

Given the content image I_c and the style image I_s , we denote the feature extracted by the VGG encoder from the content image as F_c , and the feature extracted from the style image as F_s . The size of the two feature maps is $C \times H \times W$, C is the number of channels, H is the height of the image and W is the width of the image. The paper [9] theoretically derived the form of the transformation matrix, and proved that the transformation matrix is completely determined by the covariance of the feature vector of the content image and the style image. Therefore, we can calculate the transformation matrix using the feature covariance of the content image and the style image. We define this transformation matrix as T . The fully-connected layer fc used to learn the nonlinear mapping from feature covariance to transformation matrix. The transformation matrix T is calculated by:

$$T = fc(cov(F_{cr})) \otimes fc(cov(F_{sr})) \tag{1}$$

where cov is the covariance matrix, fc is the fully-connected layer, and F_{cr} and F_{sr} will be explained in the following.

We assume that the size of the transformation matrix is $C_{matrix} \times C_{matrix}$. According to Eq. 1, obviously we need a fully-connected layer with $C_{matrix} \times C_{matrix}$ input nodes and $C_{matrix} \times C_{matrix}$ output nodes. If the C_{matrix} is large, it will lead to large memory requirements and model size. Therefore, we first use three continuous residual blocks to reduce the dimension of the input features to control the required memory capacity and the size of the model. The composition of the Residual block is defined in the paper [3], and each residual block is composed of two convolutional layers. In most image transformation networks, the input image and output image share a structure, and the residual connection can easily learn the identify function. The feature map size of the F_{cr} and F_{sr} we get from these residual blocks is $C_{matrix} \times H \times W$. From the derivation in the paper [9], we know that the terms of content and style are decoupled. Therefore, we use two independent Resblocks for content/style features.

In order for the content feature to be multiplied by the transformation matrix, we need to reduce its dimension to C_{matrix} . We use a convolutional layer to “compress” the content feature. Another convolutional layer is used to “decompress” the calculated result to the original dimension. We denote that the compressed content feature is F'_c , and the calculated result is F'_d . Then the stylized image feature F_d is calculated as shown in Eq. 2:

$$F_d = \text{uncompress}(T \otimes F'_c) + \text{mean}(F_s) \quad (2)$$

where F_d is $C \times H \times W$, which is the same as F_c and F_s . Like most existing style reconstruction methods, F_d aligns the mean and covariance statistics of the target style at the same time. So it can express the feeling of style image on the content image. F_d is decoded by the decoder corresponding to the encoder to obtain the stylized image.

3.2 Loss Functions

We define three loss functions to constrain the learning process of the transformation matrix and compress/decompress block, all of which are calculated using a pre-trained VGG-19 network. Content loss, gram loss and contextual loss constitute the final loss according to their respective weights, as shown in Eq. 3. We will briefly introduce content loss and gram loss because they are the same as the previous work [6]. Then focus on the contextual loss [15] used in our work.

$$\ell_{total} = \alpha \ell_{content} + \beta \ell_{gram} + \gamma \ell_{contextual} \quad (3)$$

Content Loss and Gram Loss. We denote the content image as I_c and the style image as I_s , and the stylized image result obtained by our proposed method is I_t . We denote the vgg loss network as ϕ . $\phi_i(x)$ denotes the feature map of the i -th layer extracted by image x from the loss network ϕ . The content loss is defined using the L_2 norm:

$$\ell_{content} = \frac{1}{C_i H_i W_i} \|\phi_i(I_c) - \phi_i(I_t)\|_2^2 \tag{4}$$

where C_i, H_i, W_i are the shape of $\phi_i(x)$ feature map. It is better to have similar feature representations than to make the content image and the stylized image exactly match each pixel.

We define gram loss as the sum of Frobenius norms between Gram matrices of VGG features at different layers:

$$\ell_{gram} = \sum_{i \in S} \|G(\phi_i(I_s)) - G(\phi_i(I_t))\|_F^2 \tag{5}$$

where S denotes a predefined set of layers and G denotes Gramian transformation. The Gramian transform can be efficiently calculated by:

$$G(x) = \frac{\varphi(x)\varphi(x)^T}{CHW} \tag{6}$$

where $\varphi(x)$ is to reconstruct the tensor with the shape of $C \times H \times W$ into $C \times HW$.

Contextual Loss. In the paper [15], a loss function was designed to measure the similarity between non-aligned images. This loss function is called contextual loss. Contextual loss can compare regions with similar semantics while considering the context of the entire image. We use it as part of the constraints of stylized images. Contextual loss and gram loss together form style loss, which can generate more appealing images. In this work, the contextual loss is defined as:

$$\ell_{contextual} = \sum_{i \in S} -\log(\text{CX}(\phi_i(I_s), \phi_i(I_t))) \tag{7}$$

where S denotes a predefined set of layers, $\text{CX}(x, y)$ is used to calculate the contextual similarity between two image features, $\text{CX}(x, y)$ is defined as:

$$\text{CX}(x, y) = \frac{1}{N} \sum_b \max_a \text{CX}_{ab} \tag{8}$$

CX_{ab} is the similarity between the feature x_a in the feature map X and the feature y_b in the feature map Y . The paper [15] uses the distance between features to measure similarity. The cosine distance d_{ab} between x_a and y_b can be calculated by:

$$d_{ab} = \left(1 - \frac{(x_a - \mu_b) \cdot (y_b - \mu_b)}{\|x_a - \mu_b\|_2 \|y_b - \mu_b\|_2} \right) \tag{9}$$

where μ_b is the mean of the features in Y . We consider features x_a and y_b as similar when $d_{ab} \ll d_{ac}, \forall c \neq b$. We first normalize the distance:

$$\tilde{d}_{ab} = \frac{d_{ab}}{\min_c d_{ac} + \epsilon} \tag{10}$$

fix $\epsilon = 1e - 5$, and then convert from distance to similarity by exponent:

$$w_{ab} = \exp\left(\frac{1 - \tilde{d}_{ab}}{h}\right) \quad (11)$$

where $h > 0$ is a band-width parameter. Finally, we normalize it:

$$CX_{ab} = w_{ab} / \sum_c w_{ac} \quad (12)$$

the above is the whole process of obtaining contextual loss. Contextual loss allows style features to be transferred between regions based on their semantic similarity, instead of globally transferring style features across the entire image. This is exactly the deficiency of the gram loss. In our work, contextual loss and gram loss complement each other to better perform style transfer.

4 Experimental Results

4.1 Implementation Details

We use MS-COCO dataset [13] as our content dataset and WikiArt dataset [16] as our style dataset to train our model. The MS-COCO dataset has approximately 80,000 images, and the WikiArt dataset has approximately 20,000 images. In the training stage, we randomly crop the training set to 256×256 pixels. But in the testing stage, our model can handle content images and style images of any size. We extract the image features of the *relu4_1* layer from the VGG-19 model and set the C_{matrix} to 32. This setting can make the model effective, and the model size and memory requirements will not be too large. We use Adam solver [8] to train our model. The initial learning rate is 10^{-4} , and the learning rate decreases as the number of iterations increases. We set a batch size of 4, and then we will discuss why this is set. The entire training stage performs 10^5 iterations, and it takes about ten hours on the NVIDIA Tesla P100 16G GPU.

In the loss function, we set $\alpha = 1.0$, $\beta = 0.02$, and $\gamma = 0.5$. The content loss is calculated based on the *relu4_1* in the pre-trained VGG19 model, which can better preserve the structure and semantic information of the content image. The style loss is calculated based on *relu1_1*, *relu2_1*, *relu3_1*, and *relu4_1* in the pre-trained VGG19 model, using multi-level style losses to capture more style information. Contextual loss is calculated based on the output of the *conv1_1*, *conv2_1*, and *conv3_1* layers in the pre-training VGG19 model, and set $h = 0.1$.

4.2 Comparison with Existing Methods

We compare our proposed method with several of the most representative methods. Comparison methods include the optimization-based method proposed by Gatys et al. [2], the meta-learning-based method proposed by Zhang et al. [23],

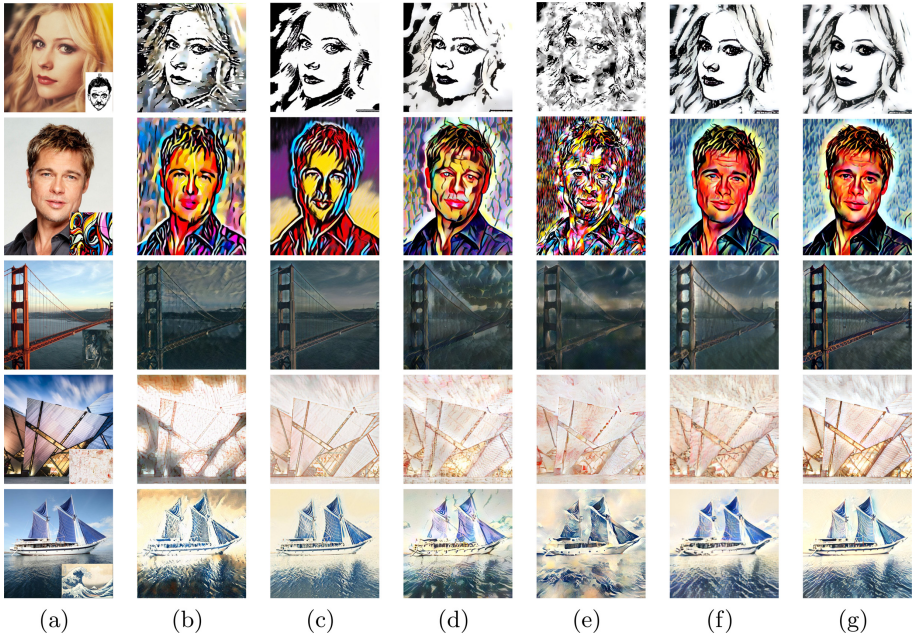


Fig. 3. Results of comparison. (a) is the content/style image. (b) is the method of Gatys et al. [2]. (c) is the method of Zhang et al. [23]. (d) is the method of Huang et al. [4]. (e) is the method of Li et al. [10]. (f) is the method of Li et al. [9]. (g) is the result of our proposed method.

the AdaIN method proposed by Huang et al. [4], the WCT method proposed by Li et al. [10] and Li et al. [9] proposed a method based on linear transformation. We use pre-trained models publicly provided by the author to test these methods. Our method can transfer arbitrary style on any size content image, and the flexibility is unquestionable. Therefore, we only compare with methods that can transfer arbitrary style. The result is shown in Fig. 3.

Quality. Figure 3 shows a qualitative comparison between our method and the existing state-of-the-art arbitrary style transfer methods. As can be seen from the figure, the stylized images generated by our method are more appealing than other methods. The method of Gatys et al. [2] has a high computational cost. The method of Zhang et al. [23] can produce attractive results, but requires a few minutes of fast-adaptation for each style. The AdaIN method [4] and the WCT method [10] are very efficient, but sacrifice the quality of stylized images. The method proposed by Li et al. [9] can achieve high-quality style transfer in a short time. But compared to our method, our method is better in terms of local details. We need to enlarge the image to realize this more clearly. For example, the eyes of the first row and the second row, the third row of bridges and the fifth row of sails. As shown in Fig. 4, we enlarge the local area of the stylized image

and compare it with the method of Li et al. [9]. From the enlarged area, we can see that the stylized image generated by our method has better structure and less distortion. Among the existing arbitrary style transfer methods, the quality of stylized images generated by our method is the best.



Fig. 4. In two images of the same style, the image on the left is the method of Li et al. [9], and the image on the right is our method. The content image and style image have been given in Fig. 3.

User Study. We conduct user study on our method and existing methods. We select 6 content images, each of which transferred 5 styles. For each combination, we present the 6 stylized images transferred by each of the above methods in random order, and ask the subject to choose the most visually appealing image. We collect 900 votes from 30 users, and Fig. 5 shows our results based on voting statistics. Our proposed method is favored among the existing arbitrary style transfer methods.

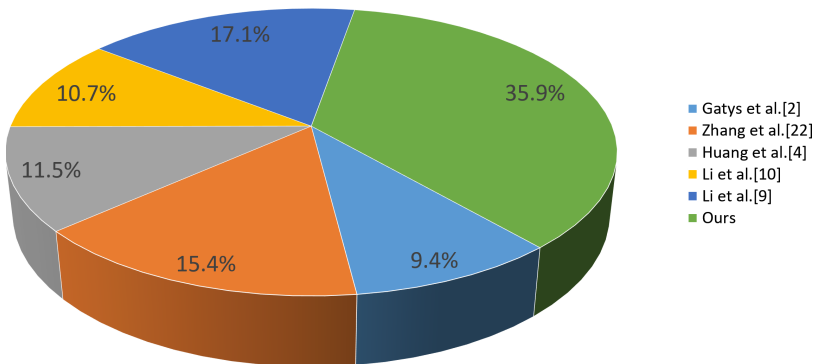


Fig. 5. Pie chart of user study results.

Table 1. Runtime performance comparison. All data is in seconds. We run the source code of these methods on a single NVIDIA Tesla P100 16G GPU. These methods use the best configuration provided by their authors.

Method	Image size		
	256×256	512×512	1024×1024
Gatys et al. [2]	19.25	64.32	130.5
Zhang et al. [23]	0.016	0.068	0.178
Huang et al. [4]	0.023	0.075	0.312
Li et al. [10]	0.982	1.124	1.132
Li et al. [9]	0.012	0.040	0.126
Ours	0.015	0.044	0.134

Speed. We compare the efficiency of our method with existing methods. We test the runtime performance of various methods when the input image size is 256×256 , 512×512 and 1024×1024 . Table 1 is the result of our test. Among all the methods, our method is in the fastest order of magnitude. Only slightly slower than the method of Li et al. [9]. The method based on meta-learning [23] is also very efficient. But we did not include the time of fast adaption, which usually takes tens of seconds. The AdaIN method [4] is fast but the quality is not satisfactory. Our method can reach 66FPS when processing 256×256 input images, which can achieve real-time style transfer.

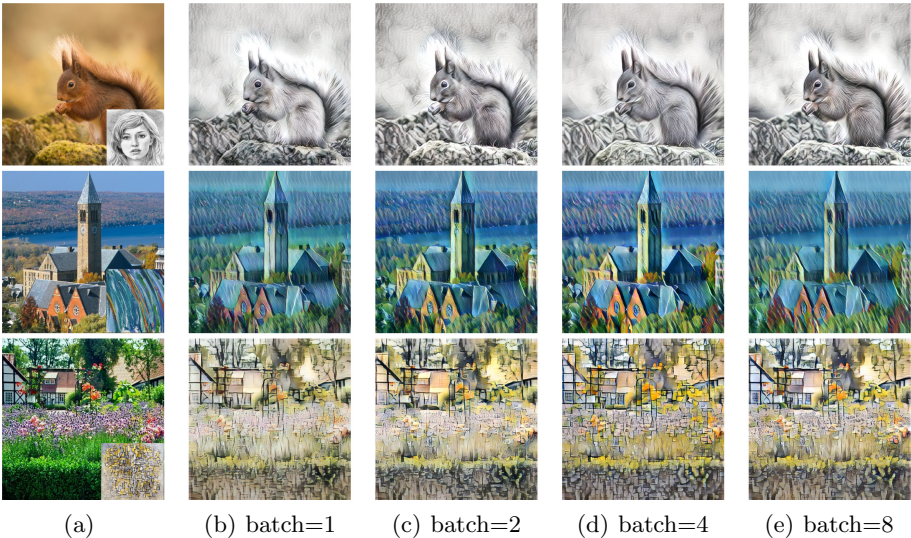


Fig. 6. Results of different batch sizes. (a) is the content/style image. (b), (c), (d) and (e) are stylized images with batch sizes of 1, 2, 4, and 8.

4.3 Impact of Batch Size

In research based on deep learning, people usually think that the larger the batch size, the faster the model converges and the shorter the training time. But in our experiments, we find that increasing the batch size may result in poor model generalization ability and performance degradation. Research [7] shows that a large batchsize converges to a sharp minimum, while a small batchsize converges to a flat minimum. The latter has better generalization ability. For the task of style transfer, the model needs to have stronger generalization ability. We conduct comparative experiments to study the influence of batch size on our method.

From Fig. 6, we can see that as the batch size increases, the quality of the stylized image gets better and better. But when the batch size is increased to 8, the generalization ability of the model degrades. For example, the first row of squirrel fur has the original color, while the second and third rows are less stylized. We think that a batch size of 4 is the most suitable batch size for our model.

5 Conclusion

This work proposes an arbitrary style transfer method based on a feed-forward network, and contextual loss is introduced in the method. The experimental results prove the effectiveness of the method proposed in this paper. Our method can generate appealing stylized images, and handles local details better than existing methods. In terms of efficiency, our method can perform real-time style transfer. It is possible to further study how to reduce the size of the model as much as possible without affecting the performance, so that it can be applied on the mobile device. We also consider further applying our method to video style transfer. This will be a very interesting research direction.

References

1. Gatys, L.A., Ecker, A.S., Bethge, M.: A neural algorithm of artistic style. arXiv preprint [arXiv:1508.06576](https://arxiv.org/abs/1508.06576) (2015)
2. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2414–2423 (2016)
3. 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)
4. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1501–1510 (2017)
5. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint [arXiv:1502.03167](https://arxiv.org/abs/1502.03167) (2015)

6. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 694–711. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_43
7. Keskar, N.S., Mudigere, D., Nocedal, J., Smelyanskiy, M., Tang, P.T.P.: On large-batch training for deep learning: Generalization gap and sharp minima. arXiv preprint [arXiv:1609.04836](https://arxiv.org/abs/1609.04836) (2016)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint [arXiv:1412.6980](https://arxiv.org/abs/1412.6980) (2014)
9. Li, X., Liu, S., Kautz, J., Yang, M.H.: Learning linear transformations for fast image and video style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3809–3817 (2019)
10. Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., Yang, M.H.: Universal style transfer via feature transforms. In: Advances in Neural Information Processing Systems, pp. 386–396 (2017)
11. Li, Y., Liu, M.Y., Li, X., Yang, M.H., Kautz, J.: A closed-form solution to photorealistic image stylization. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 453–468 (2018)
12. Liao, J., Yao, Y., Yuan, L., Hua, G., Kang, S.B.: Visual attribute transfer through deep image analogy. arXiv preprint [arXiv:1705.01088](https://arxiv.org/abs/1705.01088) (2017)
13. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: 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
14. Lu, M., Zhao, H., Yao, A., Chen, Y., Xu, F., Zhang, L.: A closed-form solution to universal style transfer. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5952–5961 (2019)
15. Mechrez, R., Talmi, I., Zelnik-Manor, L.: The contextual loss for image transformation with non-aligned data. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 768–783 (2018)
16. Nichol, K.: Painter by numbers (2016). <https://www.kaggle.com/c/painter-by-numbers>
17. Risser, E., Wilmot, P., Barnes, C.: Stable and controllable neural texture synthesis and style transfer using histogram losses. arXiv preprint [arXiv:1701.08893](https://arxiv.org/abs/1701.08893) (2017)
18. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
19. Ulyanov, D., Lebedev, V., Vedaldi, A., Lempitsky, V.S.: Texture networks: Feed-forward synthesis of textures and stylized images. In: ICML, vol. 1, p. 4 (2016)
20. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint [arXiv:1607.08022](https://arxiv.org/abs/1607.08022) (2016)
21. Yin, R.: Content aware neural style transfer. arXiv preprint [arXiv:1601.04568](https://arxiv.org/abs/1601.04568) (2016)
22. Yoo, J., Uh, Y., Chun, S., Kang, B., Ha, J.W.: Photorealistic style transfer via wavelet transforms. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 9036–9045 (2019)
23. Zhang, C., Zhu, Y., Zhu, S.C.: Metastyle: three-way trade-off among speed, flexibility, and quality in neural style transfer. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 1254–1261 (2019)
24. Zhang, H., Dana, K.: Multi-style generative network for real-time transfer. arXiv preprint [arXiv:1703.06953](https://arxiv.org/abs/1703.06953) (2017)



Enhancing Underwater Image Using Multi-scale Generative Adversarial Networks

Yujie Zhang, Peixiang Chen, Jiangyi Huang, and Yuzhong Chen[✉]

College of Mathematics and Computer Science, Fuzhou University, Fujian, China
yjzhang021@gmail.com, pxchen921@gmail.com, wbyiml@gmail.com,
yzchen1979@163.com

Abstract. Wavelength-dependent light absorption and scattering will reduce the quality of underwater images. Therefore, the characteristics of underwater images are different from those taken in natural. Low-quality underwater images affect the accuracy of pattern recognition, visual understanding, and key feature extraction in underwater scenes. In this paper, we enhance the underwater image using a multi-scale generative adversarial network with adjacent scale feature addition. Adjacent scale feature addition allows the network to more effectively capture the relevant characteristics between two image domains. The multi-scale discriminator can let the enhanced image more closer to the natural image. Our method does not rely on transmission maps and atmospheric light estimation. Experiments on a large amount of synthetic data and real data show that our method is superior to the state-of-the-art methods.

Keywords: Underwater image enhancement · Generative adversarial networks · Adjacent scale feature addition

1 Introduction

Low-quality underwater images will seriously affect underwater tasks such as underwater target detection and resource exploration. The complex imaging environment of the ocean leads to serious degradation of the underwater images acquired by the vision system. The energy attenuation occurs during the transmission of light in an underwater environment. In general, red light attenuates the fastest in water, and blue-green light attenuates the slowest, so underwater images usually look greenish. Light scattering in the water can also cause degradation of underwater images. Figure 1 shows the effect of light on underwater image degradation. The scattering is divided into forward scattering and back scattering. Forward scattering refers to the scattering phenomenon of small angle deviation that occurs when the light reflected by objects in the water is transmitted to the camera, resulting in blurred image details. When irradiating an object in the water, the impurities in the water are scattered and directly received by the camera, resulting in low image contrast. Some methods [1, 3, 11, 14, 16] reversely

derive and establish corresponding physical models based on the causes of underwater image degradation, which can achieve better results in certain scenes, such as dehazing and color correction. Then, on the one hand, the underwater scene is complex, and the corresponding parameters are also complex and changeable. It is difficult for the physical model to be fully compatible to solve this problem. On the other hand, many underwater operations require real-time operations, and these methods are difficult to meet this requirement in terms of speed.

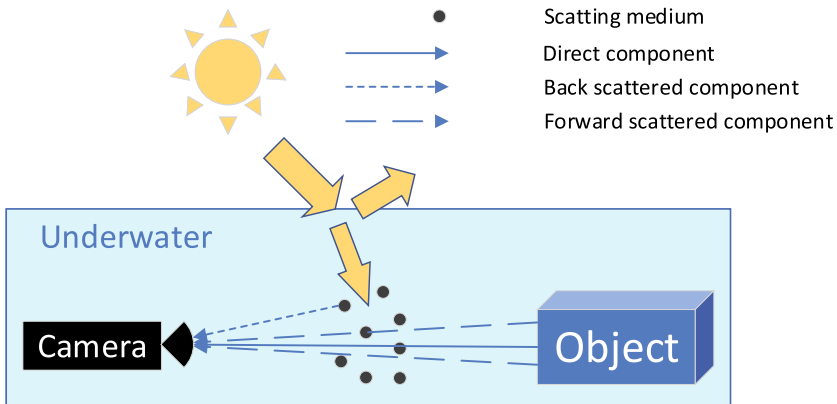


Fig. 1. The effect of light on underwater image degradation.

In recent years, deep learning networks have achieved great success in the fields of image and video understanding, image generation, and image enhancement. In view of this, a feasible alternative is to build a deep network model to enhance underwater images. Some methods [2, 7, 12, 20] based on convolutional neural networks and generative adversarial networks have been proposed for underwater image enhancement tasks. The use of large-scale data sets for deep network models is an inevitable problem. However, the acquisition of underwater data sets is difficult and costly, so it is difficult to obtain large-scale data sets. Most of the existing learning-based methods use small-scale data sets, it is difficult to capture large-scale natural variation characteristics. Therefore, it is very important to design a robust and effective underwater image enhancement model. In order to solve this problem, we design a multi-scale generative adversarial network with adjacent scale feature addition. And we analyze its practicability and effectiveness. Our contributions in this article are as follows:

1. We propose a method to add adjacent scale features, which can more effectively capture the relevant characteristics between the two image domains.
2. We improve a new attention mechanism module, which can effectively help the network pay more attention to effective information.
3. We design a multi-scale discriminator, which can better ensure the consistency of the global information and local details of the generated image and the ground truth image.

2 Related Work

The existing underwater image enhancement methods can be roughly divided into two categories, methods based on physical models and methods based on learning models. Both methods have their own advantages. This section will introduce them separately.

2.1 Underwater Image Enhancement Methods Based on Physical Model

The method based on physical model is usually to establish a mathematical model of the degradation process of underwater images, and obtains a clear underwater image by inverting the degradation process. He et al. [3] proposed the dark channel prior (DCP), which removes the haze in the image by estimating the ambient light and transmittance. However, the classic DCP has a poor recovery effect on underwater images, because the underwater images have blue or green color cast, and other channel values are small, which will cause the judgment of the degree of underwater haze to be invalid. Li et al. [11] proposed an underwater image restoration method based on blue-green channel dehazing and red channel correction. Through the expansion and modification of the dark primary color prior algorithm, the defogging algorithm is used to realize the recovery of the blue and green channels. Drews et al. [1] improved on the DCP method and basically believed that the blue and green channels are the source of underwater visual information. However, the underwater environment is complicated, and the lighting conditions are changeable, which causes these methods to have greater limitations. Peng et al. [14] proposed an underwater scene depth estimation method based on image blur and light absorption, which can estimate the underwater depth of field more accurately. And this method can be used in image formation models (IFM) to restore and enhance underwater images. Song et al. [16] proposed a fast and effective scene depth estimation model based on the underwater light attenuation prior for underwater images, and used learning-based supervised linear regression to train the model coefficients. Due to the complex underwater scenes, the physical model cannot adapt to all underwater environments and can only show good performance in specific environments. In order to adapt to the new environment, it is necessary to re-experiment and adjust the parameters, which is also its limitation.

2.2 Underwater Image Enhancement Methods Based on Learning Model

At present, methods based on learning model mainly use supervised training methods, which require a large amount of training data, including underwater degraded images and their corresponding ground truth images. However, due to the limitations of the special imaging environment, underwater ground truth images are difficult to obtain. Therefore, the data currently used is mainly synthetic data. Fabbri et al. [2] used CycleGan [19] to convert normal images into

underwater degraded images, and used the obtained data set to train an underwater image enhancement model. Li et al. [12] took normal RGB-D images and a small amount of underwater image sample sets as input, simulated the physical model of underwater degraded scenes to train the image generation network, which rendered the normal images as underwater degraded images. Obtain a paired data set, and use this data set to train an underwater image restoration network. However, RGB-D has higher requirements for cameras. Machines are generally not equipped, which has certain limitations. Li et al. [10] collected 890 real-word underwater images and enhanced them with a variety of physical models, then chose the best one as the corresponding ground truth image. FUNIE [7] proposed a conditional generative adversarial network to build a real-time enhancement model for underwater images, which can be trained using unpaired data. The training data sets currently used are usually small in scale, so it is difficult to capture a wide range of natural mutation characteristics.

3 Proposed Method

3.1 The Architecture of Our Network

Given the source domain X (underwater degraded image) and target domain Y (enhanced image), our goal is to learn the mapping $G: X \rightarrow Y$ to achieve automatic image enhancement. We use generative adversarial network for training, train a generator to learn the mapping between two image domains and a discriminator to judge whether the input image is true.

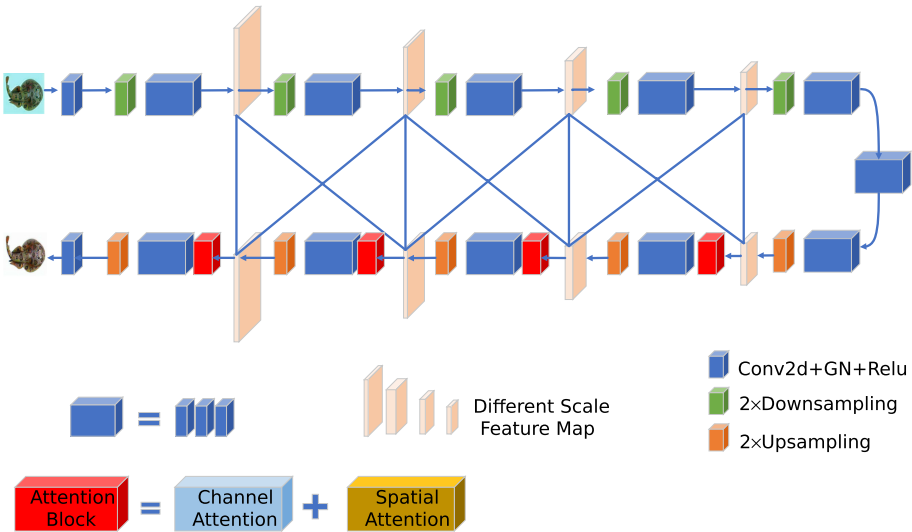


Fig. 2. The architecture of generator.

Generative Network. As shown in the Fig. 2, in order to better capture the relevant characteristics between the two image domains, we propose a generative network with adjacent scale feature addition. After the image goes through the process of down-sampling and up-sampling, information will be lost. The U-Net [15] adds the corresponding hierarchical features generated in the image up-sampling process, this method effectively reduces the loss of information. Guo et al. [20] used the dense block to render more details in the generator, which will add a lot of computation. We propose a method to add together features of the same scale and adjacent scales generated in the image up-sampling process. Proved in experiments that it is very suitable for underwater image enhancement.

Attention Block. In recent years, the attention mechanism has been widely used in the field of computer vision. Hu et al. [5] proposed a channel attention module for network channels (SENet). And Woo et al. [18] proposed an attention mechanism module named CBAM that combines space and channel. Wang et al. [17] improved on the basis of SENet and proposed ECANet. We replace the channel attention module in the CBAM with ECANet. We use this module after feature addition, which can effectively help the network pay more attention to effective information.

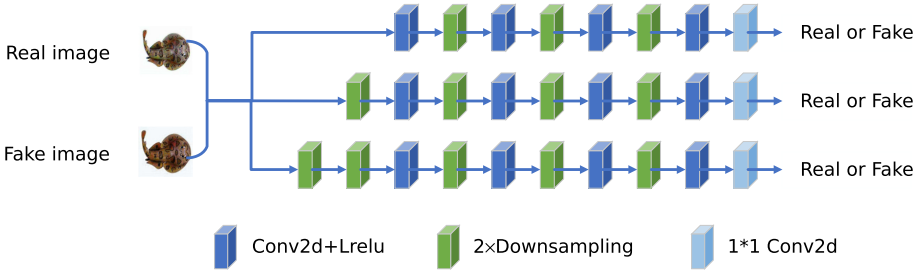


Fig. 3. The architecture of multi-scale discriminator.

Multi-scale Discriminator. As shown in the Fig. 3, We design a multi-scale discriminator to judge whether the input image is a true image. The network includes three sub-networks, namely D_1 , D_2 , and D_3 . The three networks have the same network structure and the input is a different size of the same image. The network adopts the PatchGAN [8] architecture, which divides the image into blocks and can obtain high-frequency features such as texture style more effectively. Different scale discriminators can obtain different receptive fields of the image. The large scale discriminator pays more attention on global information, and the small scale discriminator pays more attention on detail information. The multi-scale discriminator can make the generated images have higher quality.

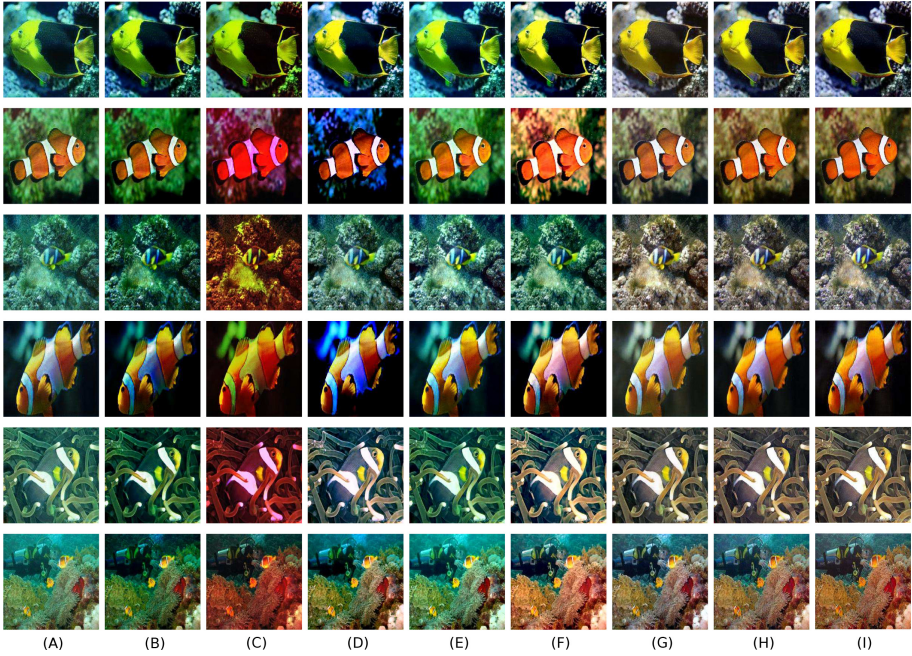


Fig. 4. Qualitative comparisons on synthetic underwater images. (A) Original images. Restored results using (B) UDCP [1] (C) GB [11] (D) IBLA [14] (E) RGHS [6] (F) ULAP [16] (G) FUNIE [7] (H) Our method (I) Ground Truth.

3.2 Overall Loss Function

In order to optimize the network, as shown in the Eq. 1, we use three loss functions, the adversarial loss L_A , the style loss L_S , and the image loss L_I . Image loss is used to constrain the global structural consistency between the generated image and the reference image. Adversarial loss and style loss are used to enhance the fine features of the image. λ_1 , λ_2 and λ_3 are the weight coefficient.

$$L = \lambda_1 L_A + \lambda_2 L_S + \lambda_3 L_I \quad (1)$$

Adversarial Loss. We use the least squares GAN loss [13] to train the generator and discriminator. As shown in the Eq. 2, where G is the generator, D is the discriminator, z is the underwater degraded image, and x is the corresponding reference truth image, the value of a , b , c is 0, 1, 1.

$$\begin{aligned} \min_D L(D) &= E_{x \sim p_x} (D(x) - b)^2 + E_{z \sim p_z} (D(G(z)) - a)^2 \\ \min_G L(G) &= E_{z \sim p_z} (D(G(z)) - c)^2 \end{aligned} \quad (2)$$

Since we use a multi-scale discriminator, the loss becomes a multi-scale learning problem. Our goal is to optimize the following formula:

$$\min_G \min_{D_1, D_2, D_3} \sum_{k=1,2,3} L(G, D_k) \quad (3)$$

Style Loss. We design the style loss with the perceptual loss [9] which is widely used in the task of style transfer. First, use the pre-trained VGG network to extract the features of different network layers, and calculate the gram matrix. It is defined as follows:

$$G_j^\phi(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'} \quad (4)$$

where j represents the j_{th} layer of the network, and $C_j H_j W_j$ represents the size of the feature map of the j_{th} layer. Then calculate the squared Frobenius norm of the difference between the output \hat{y} and target image y . It is defined as follows:

$$\ell_{style}^{\phi,j}(\hat{y}, y) = \left\| G_j^\phi(\hat{y}) - G_j^\phi(y) \right\|_F^2 \quad (5)$$

Image Loss. Calculating the absolute error value between the output enhanced image X and the ground-truth image Y . It is defined as follows:

$$L_I = \|X - Y\|_1 \quad (6)$$

4 Experiment

4.1 Experiment Settings

In this section, we verify the effectiveness of the method proposed in this paper on synthetic data set and real-word data set. We compare our method with the following methods: UDCP [1], IBLA [14], GB [11], ULAP [16], RGHS [6] and FUNIE [7].

Dataset. We used the synthetic dataset proposed by Fabbri et al. [2] and the real-word dataset (UIEBD) collected by Li et al. [10]. The synthetic dataset has 6128 pairs of paired data. We randomly select 5000 pairs of data as training data, and use the remaining 1128 pairs of data as the test sets. The UIEBD has 890 real underwater scene data, we randomly selected 800 pairs of data as training data, and use the remaining 90 pairs of data as the test sets. We compare our method with other methods on these two data sets.

Training Details. During training, we adopt Adam optimizer with a batch size of 4, and set a learning rate as 0.001, the exponential decay rates as $(\beta_1, \beta_2) = (0.5, 0.999)$. The hyperparameters of loss function are set as $\lambda_1 = 1$, $\lambda_2 = 1$, $\lambda_3 = 2$. We implement our model with the PyTorch framework and an Nvidia GTX 2080 Ti GPU.

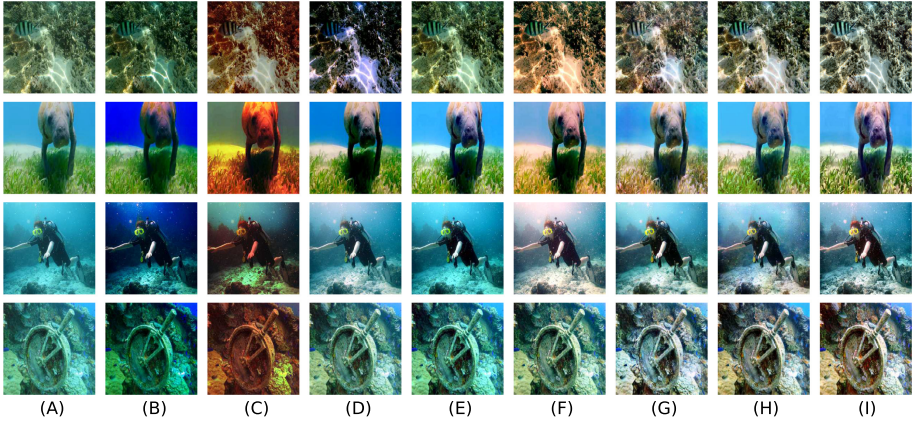


Fig. 5. Qualitative comparisons on real underwater images. (A) Original images. Restored results using (B) UDCP [1] (C) GB [11] (D) IBLA [14] (E) RGHS [6] (F) ULAP [16] (G) FUNIE [7] (H) Our method (I) Ground Truth.

Table 1. Quantitative comparison for average PSNR and SSIM values on different datasets.

Dataset		UDCP [1]	IBLA [14]	GB [11]	ULAP [16]	RGHS [6]	FUNIE [7]	Ours
Synthetic dataset	PSNR	18.23	19.00	19.17	22.51	21.80	23.18	24.10
	SSIM	0.65	0.64	0.67	0.71	0.73	0.76	0.77
Real dataset	PSNR	14.87	18.02	15.49	17.94	18.27	18.09	20.36
	SSIM	0.54	0.68	0.60	0.70	0.77	0.74	0.77

4.2 Qualitative Evaluations

As shown in Fig. 4 and Fig. 5, we compare our method with the classic method and the excellent methods of recent years of underwater image enhancement tasks on synthetic data sets and real data sets. Visually, these methods have more or less chromatic aberration and loss of details, the image enhanced by our method is the closest to the reference image.

4.3 Quantitative Evaluations

We use quality evaluation PSNR, SSIM [4] to evaluate the performance of our method.

The PSNR approximates the reconstruction quality of a generated image x compared to its ground truth y based on their Mean Squared Error (MSE) as follows:

$$PSNR(x, y) = 10 \log_{10} [255^2 / MSE(x, y)] \quad (7)$$

The SSIM compares the image patches based on three properties: luminance, contrast, and structure. It is defined as follows:

$$SSIM(x, y) = \left(\frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \right) \left(\frac{2\sigma_{xy} + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \right) \quad (8)$$

As shown in Table 1, from the average value of the results, our method is superior to other methods.

4.4 Ablation Study

In order to better prove the effectiveness of our method architecture, We do the following two comparative experiments: 1. Remove the adjacent scale feature addition (ASFA); 2. Replace the attention module with the original CBAM module. As shown in Table 2, the results show that the method of adjacent scale feature addition and the improved attention module are effective for underwater image enhancement.

Table 2. The result of ablation study.

Dataset	Method	PSNR	SSIM
Synthetic dataset	Network without ASFA	23.53	0.74
	Network with CBAM	23.96	0.75
	Ours	<i>24.10</i>	<i>0.77</i>
Real dataset	Network without ASFA	19.85	0.75
	Network with CBAM	20.17	0.76
	Ours	<i>20.36</i>	<i>0.77</i>

5 Conclusion

The quality of underwater image is essential for pattern recognition, visual understanding, and key feature extraction in underwater scenes. In this paper, we enhance the underwater image using a multi-scale generative adversarial network with adjacent scale feature addition. This method does not rely on transmission maps and atmospheric light estimation. We turn the underwater image enhancement problem into an image-to-image conversion problem. Experimental results on both the synthesis dataset and the real-world dataset demonstrate that the proposed method achieves the best performance of underwater enhancing in both the quantitative and qualitative evaluations.

References

1. Drews, P., Nascimento, E., Moraes, F., Botelho, S., Campos, M.: Transmission estimation in underwater single images. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, pp. 825–830 (2013)
2. Fabbri, C., Islam, M.J., Sattar, J.: Enhancing underwater imagery using generative adversarial networks. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 7159–7165. IEEE (2018)
3. He, K., Sun, J., Tang, X.: Single image haze removal using dark channel prior. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(12), 2341–2353 (2010)
4. Hore, A., Ziou, D.: Image quality metrics: PSNR vs. SSIM. In: 2010 20th International Conference on Pattern Recognition, pp. 2366–2369. IEEE (2010)
5. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018)
6. Huang, D., Wang, Y., Song, W., Sequeira, J., Mavromatis, S.: Shallow-water image enhancement using relative global histogram stretching based on adaptive parameter acquisition. In: Schoeffmann, K., et al. (eds.) MMM 2018. LNCS, vol. 10704, pp. 453–465. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-73603-7_37
7. Islam, M.J., Xia, Y., Sattar, J.: Fast underwater image enhancement for improved visual perception. *IEEE Robot. Autom. Lett.* **5**(2), 3227–3234 (2020)
8. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134 (2017)
9. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9906, pp. 694–711. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46475-6_43
10. Li, C., Guo, C., Ren, W., Cong, R., Hou, J., Kwong, S., Tao, D.: An underwater image enhancement benchmark dataset and beyond. *IEEE Trans. Image Process.* **29**, 4376–4389 (2019)
11. Li, C., Quo, J., Pang, Y., Chen, S., Wang, J.: Single underwater image restoration by blue-green channels dehazing and red channel correction. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1731–1735. IEEE (2016)
12. Li, J., Skinner, K.A., Eustice, R.M., Johnson-Roberson, M.: Watergan: unsupervised generative network to enable real-time color correction of monocular underwater images. *IEEE Robot. Autom. Lett.* **3**(1), 387–394 (2017)
13. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2794–2802 (2017)
14. Peng, Y.T., Cosman, P.C.: Underwater image restoration based on image blurriness and light absorption. *IEEE Trans. Image Process.* **26**(4), 1579–1594 (2017)
15. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28

16. Song, W., Wang, Y., Huang, D., Tjondronegoro, D.: A rapid scene depth estimation model based on underwater light attenuation prior for underwater image restoration. In: Hong, R., Cheng, W.-H., Yamasaki, T., Wang, M., Ngo, C.-W. (eds.) PCM 2018. LNCS, vol. 11164, pp. 678–688. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-00776-8_62
17. Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., Hu, Q.: Eca-net: efficient channel attention for deep convolutional neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11534–11542 (2020)
18. Woo, S., Park, J., Lee, J.Y., So Kweon, I.: Cbam: convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV), pp. 3–19 (2018)
19. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2223–2232 (2017)
20. Guo, Y., Li, H., Zhuang, P.: Underwater image enhancement using a multiscale dense generative adversarial network. *IEEE J. Oceanic Eng.* **45**, 862–870 (2019)



Inferring Prerequisite Relationships Among Learning Resources for HPC Education

Run Wu^{1,2}, Chenyu Luo^{1,2}, Miao Hu^{1,2}, and Di Wu^{1,2,3}(✉)

¹ School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China

wudi27@mail.sysu.edu.cn

² Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, China

³ Peng Cheng Laboratory, Shenzhen, China

Abstract. For online HPC (High-Performance Computing) education, identifying the prerequisite relationships among different HPC learning resources is important for generating learning paths and personalized course recommendations for students. Previously, such prerequisite relationships were often created by domain experts manually. In this paper, we propose a novel framework to infer both concept and learning resource prerequisite relationships and integrate such a framework into an online HPC education platform. We first construct a bi-directional long short-term (BiLSTM) neural network with attention mechanism to automatically mine the latent semantic features from the formal description of concepts. Then, we input two kinds of features into a fully connected neural network for classification and obtain the relationships among concepts. Considering the asymmetry and directivity of the prerequisite relation, we adopt a network embedding model to learn the vector representations of each concept as the prerequisite and subsequent roles. The concept vectors are weighted and summed to generate the feature vectors of learning resources. For a given pair of learning resources, their feature vectors are input into a classifier that outputs the result of relation classification. Finally, we conduct a series of experiments on two large benchmark datasets. The experimental results show that our method outperforms existing methods with significant improvements.

Keywords: Learning resources · Concept prerequisite relationship · Network embedding

1 Introduction

The demands of online education have been showing a continuous and rapid growth in the last decade driven by the development of the Internet. A large number of online education resources can be quickly obtained from various sources,

such as Massive Open Online Courses (MOOC) and free technical blogs. However, most MOOC websites simply tag and categorize learning resources by keywords without providing knowledge dependency, which makes it challenging for learners to find a proper learning path in such a huge course space.

Generally, learning resources as defined as any resources that can be used to accomplish learning goals (e.g., a blog or a video). Concepts are the basic units in learning resources that refer to professional terms in specific fields. The prerequisite relation is a natural dependence among knowledge concepts when people learn, organize, and apply knowledge [4]. For example, before a learner starts to study machine learning, he may need to learn the related knowledge of linear algebra and probability theory in advance. Besides, learners may waste a lot of time learning overlapping content in similar courses. Therefore, if the massive learning resources are organized into a reasonable logical sequence based on the prerequisite relationships, it can help learners with different backgrounds to explore knowledge more easily.

There have been a few efforts aiming to learn prerequisite relations for concepts and learning resources. For example, CGL [16] was a supervised framework to learn course prerequisites by constructing a universal concept graph. In contrast, CPR-Recover [4] focused on recovering concept prerequisites from course dependencies. MOOC-RF [9] was the first method considering MOOC data and they proposed several kinds of features. Recently, PREREQ [10] used pairwise-link LDA model [8] to learn concept representations and predict concept relations with a Siamese network [1]. Despite the related work, challenges remain: 1) How to represent a learning resource more efficiently. 2) How to automatically extract deep features from the available corpus? 3) How to ensure accuracy even when the training data is insufficient?

To address the above challenges, we propose CPRI, a deep learning-based method for Concept Prerequisite Relationship Inferring. By exploiting the structured and unstructured information of the Wikipedia corpus, we construct rich heterogeneous features that help relation inference. Besides, CPRI uses a BiLSTM network with attention mechanism to extract the deep semantic features from the description text of concepts. The two kinds of features are then fused and input into a fully connected neural network that outputs the result of whether there exists a prerequisite relation between two concepts. Next, we design an algorithm named CPNE to identify the prerequisite relationships among learning resources. CPNE first learns the vector representation of the concepts with a multi-role network embedding method. The concept vectors are then weighted by TF-IDF and summed up to generate the learning resource vectors that are passed to a Random Forest classifier to identify the relationships between the learning resources. Furthermore, we integrate our framework into an online HPC education platform [6, 15].

In summary, our contributions in this paper can be concluded as below:

- For identifying the concept prerequisites, we propose CPRI that combines feature construction and deep learning to improve the accuracy. We construct

rich heterogeneous features based on Wikipedia and use a BiLSTM network with attention mechanism to automatically extract deep semantic features.

- In view of the directional and asymmetry of the prerequisite relationship, we propose a multi-role representation network embedding method called CPNE to solve the problem that the conventional network embedding methods only learn a single vector representation for each node.
- Our experiments are based on two benchmark datasets from universities and MOOCs. The experimental results show that the CPRI and CPNE significantly outperform the existing approaches on the benchmark datasets.

The rest of the paper is structured as follows. Section 2 reviews the existing related works. Then, we introduce the features extracted from Wikipedia and the model of CPRI in Sect. 3. The details of the network embedding algorithm CPNE are proposed in Sect. 4. In Sect. 5, we conduct a series of experiments to provide performance evaluation. We conclude this work in Sect. 6.

2 Related Work

Learning prerequisite relations from educational data has aroused tremendous attentions in recent years. Multiple works have explored the design of data-driven methods for automatically mining prerequisite relations. For curriculum design, *Vuong et al.* [13] developed a method for finding prerequisites within a curriculum based on the automatic analysis of assessment data which records the performance of students. Some works exploited Wikipedia data [2, 11], where both Wikipedia article content and their linkage structures were utilized. To measure prerequisites between Wikipedia concepts, *Liang et al.* [2] proposed a link-based metric called *ReDf*. Based on the Wikipedia features, *Wang et al.* [14] designed a framework to extract a concept map from textbooks which jointly extracts key concepts and learns their prerequisite relations. *Liang et al.* [3] further investigated the applicability of active learning to the concept prerequisite learning on the concept map dataset from [14]. For automatic curriculum planning, *Yang et al.* [16] developed CGL, a supervised learning-based framework to map courses from different universities into a universal concept space and optimize a concept relation matrix that can be used to predict relations between unseen course pairs. With a similar data setting as CGL, CPR-Recovery focused on recovering concept prerequisite relations from course dependencies. Combining the learning-based and recovery-based methods, *Lu et al.* [5] proposed a domain-specific concept extraction approach and an iterative prerequisite relation learning approach. *Pan et al.* figured out what kinds of information in MOOCs can be used and defined one semantic feature, three contextual features, and three structural features to help prerequisites inferring. Recently, PREREQ [10] learned a latent representation of concepts with pairwise-link LDA model [8] and train a Siamese network [1] to infer unknown concept prerequisites from university courses and MOOC video playlist data.

3 Concept Prerequisites Inference

Let C be the set of all concepts of interest, that is assumed to be fixed and known in advance. For a given concept pair (a, b) , we aim to find a mapping $M : C^2 \rightarrow \{0, 1\}$, where M is defined as:

$$M(a, b) = \begin{cases} 1, & \text{if } a \Rightarrow b \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

where $a \Rightarrow b$ means a is a prerequisite concept of b .

We associate each concept in C with a unique Wikipedia article. Wikipedia is the largest encyclopedia website, covering almost all areas of knowledge, including more than 200 language versions. Wikipedia contains rich structured and unstructured information that enables us to extract high-quality semantic features. For a concept pair (a, b) , we denote the corresponding Wikipedia articles as A and B , respectively. Based on the Wikipedia corpus, the extracted features are mainly divided into artificial features and deep features.

3.1 Artificial Features

Each Wikipedia article contains a formal definition and a detailed explanation of the concept. An article includes four basic attributes: title, abstract, body, and category. We extract the following textual features based on the text information:

- (#1#2) $|A_s| / |B_s|$, the number of words in A/B 's abstract.
- (#3#4) $|A| / |B|$, the number of words in A/B 's body.
- (#5#6) Whether A_t/B_t appears in B_t/A_t , where A_t/B_t represents the title of A/B .
- (#7#8) Whether A_t and B_t have overlapping words.
- (#9#10) The number of times A_t/B_t appears in B_s/A_s .
- (#11#12) The number of times A_t/B_t appears in B/A .
- (#13) $Sim(v_a, v_b)$, the cosine similarity of v_a and v_b . v_a and v_b are the Word2vec vectors of a and b . The cosine similarity is calculated as follows:

$$Sim(v_a, v_b) = \frac{v_a \cdot v_b}{\|v_a\| \times \|v_b\|} \quad (2)$$

In Wikipedia, an article usually contains a lot of links that refer to other articles. Users may be able to acquire background knowledge from the related articles. Therefore, we can find out the prerequisites between Wikipedia concepts with the links. Based on the link structure, we define the following features:

- (#14#15) $|In(A)| / |In(B)|$, the in degree of A/B , where $In(A)/In(B)$ is the in-links of A/B , that is, the set of articles that refer to A/B .
- (#16#17) $|Out(A)| / |Out(B)|$, the out degree of A/B , where $Out(A)/Out(B)$ is the out-links of A/B , that is, the set of articles that A/B refers to.
- (#18#19) $|Out(A) \cap Out(B)|$, the common articles that A and B refer to.

- (#20#21) $Link(A, B) / Link(B, A)$, the number of links that A/B refers to B/A .
- (#22#23) $\sum_{c \in Out(A)} Link(c, B) / \sum_{c \in Out(B)} Link(c, B)$, the number of articles in $Out(A)/Out(B)$ that refer to B/A .

3.2 Deep Features

Despite the features constructed in Sect. 3.1, these features sometimes appear high sparsity, leading to inaccurate classification results. Therefore, we train a neural model named CPRI (Concept Prerequisite Relationship Inferring) by using BiLSTM units with attention mechanism to automatically identify the key content for the prerequisite inferring. By assigning higher attention weight to keywords, the model can mine the deep semantic features from the article text. As illustrated in Fig. 1, the CPRI model includes six layers.

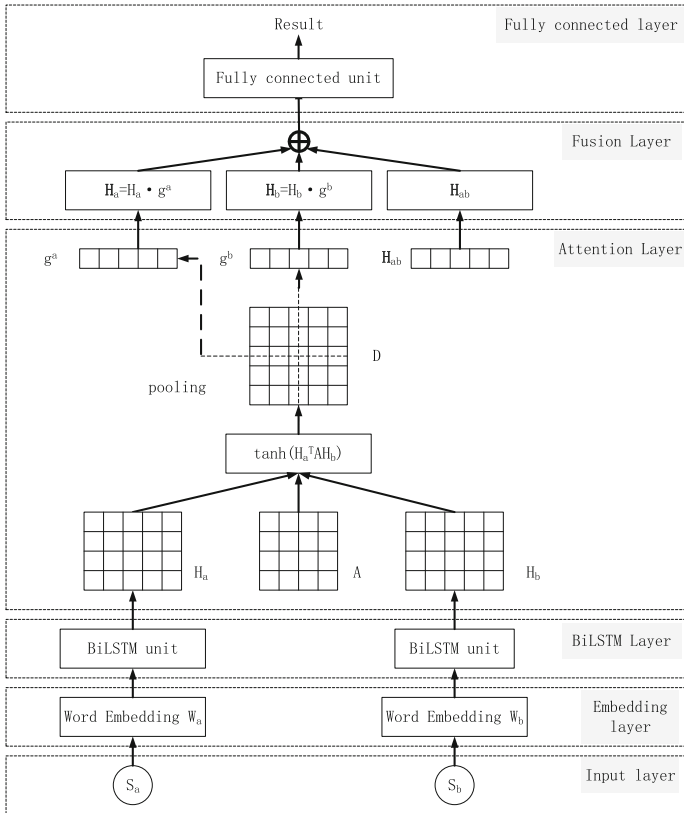


Fig. 1. CPRI model

- 1) The input layer receives the first K words of the article text of two concepts and converts each word into an index number in the dictionary V . The index sequence of concept a is indexed as $S_a = \{w_k\}_{k=1}^K$.
- 2) The word embedding layer initializes a word embedding matrix $W \in \mathbb{R}^{d \times |V|}$ with pre-trained Word2vec vectors, where d indicates the word embedding size. By looking up W , S_a is converted into an embedding matrix $W_a = \{\mathbf{w}_k\}_{k=1}^K$.
- 3) The BiLSTM layer computes each word embedding by reading W in two opposite directions and generates two vectors, i.e. a forward vector \overrightarrow{h}_k and a backward vector \overleftarrow{h}_k . Note that the size of \overrightarrow{h}_k and \overleftarrow{h}_k are determined by the number of hidden neurons, we set both the sizes of the forward and backward hidden cells equals to the word embedding size. Let h_k be the element-wise sum of \overrightarrow{h}_k and \overleftarrow{h}_k , and $H = \{h_k\}_{k=1}^K$ denotes the matrix formed by the output vectors.
- 4) The attention layer aims to identify the key content for prerequisite inferring and assign different weights. We introduce an attention matrix $A \in \mathbb{R}^{d \times d}$ to multiply H_a and H_b , and operate a non-linear transformation over the product to generate a correlation matrix $D \in \mathbb{R}^{K \times K}$ as follows:

$$D = \tanh(H_a^T \mathbf{A} H_b) \tag{3}$$

where each element $D_{i,j}$ in D represents the pair-wise correlation score between the i -th hidden vector of H_a and the j -th hidden vector of H_b . Then, we perform mean-pooling operations along rows and columns of D to get the importance vectors g_a and $g_b \in \mathbb{R}^K$. After that, we multiply H_a with g_a to get the deep feature \mathbf{H}_a and we get \mathbf{H}_b in the same way.

- 5) In the fusion layer, the artificial features \mathbf{H}_{ab} are normalized and concatenated with \mathbf{H}_a and \mathbf{H}_b to generate the eventual feature as the input of the next layer.
- 6) The fully connected layer has two hidden layers that contain 128 neurons. A rectified linear unit (ReLU) is applied as the activation function. Inferring the prerequisite relationship between two concepts is actually solving a binary classification problem, thus the output layer consists of two neurons whose outputs represent the probability of whether a being $a \Rightarrow b$ or not. In the training phase, we use the cross-entropy loss function and train the model with the stochastic gradient descent (SGD) method.

4 Learning Resource Prerequisites Identification

The result of concept prerequisites from CPRI is denoted as $L = \{(a, b) | a, b \in C, a \Rightarrow b\}$. Obviously, the sets C and L form a concept network with concepts as nodes and prerequisite relations as directional edges. Based on the network, we propose a network embedding (NE)-based method to learn the low-dimensional vector representations of concepts that imply the prerequisite relationship.

The conventional NE methods generally use a single vector to represent a node. However, these methods are insufficient to solve the prerequisite network due to the asymmetry and directivity. To address the problem, we develop CPNE, a novel NE model that learns two vector representation of each concept as the prerequisite and subsequent roles to capture the characteristics of the prerequisite relationship. We extend each pair (a, b) in L by adding a label $\varepsilon_{ab} \in \{-1, 1\}$ and obtain a tuple (a, b, ε_{ab}) . ε_{ab} is set to 1 if $a \Rightarrow b$ and 0, otherwise. Due to the asymmetry, a new tuple $(b, a, -1)$ is added to L if the tuple $(a, b, 1)$ exists.

For a concept a , let v_a^{out} and v_a^{in} denote the vector representation when it plays a prerequisite and subsequent roles, respectively. If $a \Rightarrow b$, CPNE tends to minimize the distance of v_a^{out} and v_b^{in} , and concurrently enlarge the distance of v_a^{in} and v_b^{out} . This enables the vectors to learn the characteristics of the prerequisite relationship.

CPNE applies a neural network to embed the concepts. Different from the existing methods, CPNE initializes two parameter matrices W_{out} and $W_{in} \in \mathbb{R}^{|C| \times d}$ for the prerequisite and subsequent concepts, respectively. The embedding process is described in Algorithm 1.

Algorithm 1: Concept Prerequisite Network Embedding Algorithm

Input: concept set C , prerequisite relation set L , training batch size m , iteration times $iter$, number of negative samples K , learning rate λ

Output: Parameter matrices $W^{out}, W^{in} \in \mathbb{R}^{|C| \times d}$

Initialize Parameter matrices W^{out}, W^{in} ; **for** $i = 1$ **to** $iter$ **do**

randomly sample m tuples from L as the training data T_i

forall (a, b, ε_{ab}) **in** T_i **do**

look up the vector v_a^{out} from W^{out}

look up the vector v_b^{in} from W^{in}

compute the gradient of v_a^{out} according to 5

and update v_a^{out}

compute the gradient of v_b^{in} according to 5

and update v_b^{in}

for $k = 1$ **to** K **do**

randomly sample a concept c if $(a, c, 1) \notin L$

look up the vector v_c^{in} from W^{in}

compute the gradient of v_a^{out} according to 7

and update v_a^{out}

compute the gradient of v_b^{in} according to 8

and update v_c^{in}

end

end

end

The goal of CPNE is to optimize the following function:

$$J = \sum_{(a,b,\varepsilon_{ab}) \in T_i} [-\log \sigma(\varepsilon_{ab} \cdot v_a^{out} \cdot v_b^{in}) - \sum_{k=1}^K \log \sigma(-v_a^{out} \cdot v_c^{in})] \quad (4)$$

where σ represents a Sigmoid function, T_i is the training batch in the i -th iteration, K is the number of negative samples [7], and v_c^{in} is the vector of a randomly sampled concept c .

In the training phase, CPNE adopts the SDG method to update the vectors. For a tuple (a, b, ε_{ab}) , the gradients of v_a^{out} and v_b^{in} are calculated as follows:

$$\frac{\partial J_{a,b}}{\partial J_{v_a^{out}}} = -\varepsilon_{ab} \cdot v_b^{in} \cdot (1 - \sigma(\varepsilon_{ab} \cdot v_a^{out} \cdot v_b^{in})), \quad (5)$$

$$\frac{\partial J_{a,b}}{\partial J_{v_b^{in}}} = -\varepsilon_{ab} \cdot v_a^{out} \cdot (1 - \sigma(\varepsilon_{ab} \cdot v_a^{out} \cdot v_b^{in})). \quad (6)$$

In the negative sampling phase, for a tuple $(a, c, -1)$, the gradients of v_a^{out} and v_c^{in} are calculated as:

$$\frac{\partial J_{a,c}}{\partial J_{v_a^{out}}} = v_c^{in} \cdot (1 - \sigma(-v_a^{out} \cdot v_c^{in})) = v_c^{in} \cdot \sigma(v_a^{out} \cdot v_c^{in}), \quad (7)$$

$$\frac{\partial J_{a,c}}{\partial J_{v_c^{in}}} = v_a^{out} \cdot (1 - \sigma(-v_a^{out} \cdot v_c^{in})) = v_a^{out} \cdot \sigma(v_a^{out} \cdot v_c^{in}). \quad (8)$$

We use the concept vectors output by CPNE to generate the vector representation for learning resources. Using the text information, i.e., titles and introductions, we first represent each learning resource with a bag-of-words of concepts instead of using all words in the text. We then calculate the weights of the concepts towards each learning resource by term frequency-inverse document frequency (TF-IDF) and multiply them with the corresponding concept vectors. The weighted concept vectors are then summed up to form the learning resource vectors. After that, we obtain the vector representations of each learning resource as the prerequisite and subsequent roles. Given a pair of learning resources, we input their vectors into a Random Forest classifier to identify their prerequisite relationship.

5 Performance Evaluation

5.1 Datasets

We use two benchmark datasets towards the Computer Science (SC) domain, one is the University Course Dataset [10] that contains real data collected from 11 U.S universities, the other is the NPTEL MOOC Dataset [9] whose data are crawled from a MOOC corpus. The detailed descriptions of two datasets are listed in Table 1.

Table 1. Statistics of the two datasets

Dataset	Learning resources	Learning resource pairs	Learning resource prerequisites	Concepts	Concept pairs	Concept prerequisites
University	568	3012	1276	340	2085	806
NEPTL MOOC	382	2251	861	345	2322	929

5.2 Baselines

We compare the performance of CPRI with three existing methods:

- PREREQ [10], a supervised method using a pairwise-link LDA model to obtain vector representations of concepts and a Siamese network to predict unknown concept prerequisites.
- MOOC-RF [9], a graph-based method that proposes multiple kinds of features on MOOCs data and classifies the concept relations with a Random Forest classifier.
- RefD [2], a metric for measuring the probability of one Wikipedia concept being a prerequisite to another one based on the Wikipedia links.

In the part of learning resources, we mainly compare CPNE with CGL [16]. CGL represents a learning resource with a bag-of-word model of concepts. Based on the labeled prerequisite data of learning resources, it constructs and optimizes a matrix of concept prerequisites. Through the matrix, CGL calculated the prerequisite score of the unknown learning resource pairs. To verify the efficiency of the vector representation of CPNE, we use another two methods to generate concept vectors as a comparison, i.e., pairwise-link LDA (PL-LDA) [10] and CANE [12].

5.3 Performance Comparison

We split the datasets into training sets and test sets. For each dataset, we randomly sample 20%, 30%, and 40% tuples for training, and the remaining for testing. All experiments are evaluated over 5 train-test splits.

5.4 Results for Concept Part

For CPRI, we choose the length of concept text $K = 300$ and the size of Word2vec vector $d = 100$. The experimental results of the concept part on the two datasets are shown in Table 2 and Table 3. We can find that CPRI consistently performs better than the existing approaches over different groups of data. CPRI achieves the best precision, recall and, F-score values on both the University Dataset and MOOC Dataset. Even when only 20% of the available labeled data is used for training, CPRI maintains good performance.

Table 2. Results of the concept prerequisite inference on the University Course Dataset

Metrics	Precision			Recall			F1-score		
	20	30	40	20	30	40	20	30	40
Training data(%)	20	30	40	20	30	40	20	30	40
RefD	0.466	0.477	0.502	0.486	0.552	0.463	0.469	0.512	0.482
MOOC-RF	0.588	0.604	0.631	0.627	0.625	0.636	0.607	0.614	0.633
PREREQ	0.574	0.610	0.632	0.692	0.706	0.693	0.628	0.654	0.661
CPRI	0.844	0.832	0.853	0.826	0.850	0.865	0.835	0.840	0.854

Table 3. Results of the concept prerequisite inferring on the MOOCs Dataset

Metrics	Precision			Recall			F1-score		
	20	30	40	20	30	40	20	30	40
Training data(%)	20	30	40	20	30	40	20	30	40
RefD	0.503	0.489	0.532	0.506	0.522	0.513	0.504	0.505	0.522
MOOC-RF	0.608	0.622	0.638	0.564	0.598	0.611	0.585	0.610	0.624
PREREQ	0.556	0.594	0.607	0.732	0.726	0.757	0.632	0.653	0.674
CPRI	0.828	0.824	0.839	0.845	0.873	0.868	0.836	0.848	0.853

5.5 Results for Learning Resource Part

For CPNE, we set the training batch size as 64 and the total iterations $iter = 200$. The number of negative samples K is set to 10 and the learning rate λ is set to 0.001. The experimental results of the learning resource part are illustrated in Table 4 and Table 5. As shown, CPNE outperforms the baselines significantly. The results demonstrate that CPNE is capable to capture the characteristics of the network structure and maintain the asymmetry and directivity well. Therefore, the vectors learned by CPNE are more accurate to represent the prerequisite relationship.

Table 4. Results of the learning resource prerequisite inferring on the University Course Dataset

Metrics	Precision			Recall			F1-score		
	20	30	40	20	30	40	20	30	40
Training data(%)	20	30	40	20	30	40	20	30	40
CGL	0.627	0.635	0.644	0.671	0.668	0.689	0.648	0.651	0.666
PL-LDA	0.536	0.545	0.566	0.602	0.634	0.623	0.567	0.586	0.593
CANE	0.672	0.668	0.697	0.659	0.694	0.710	0.665	0.681	0.703
CPNE	0.733	0.750	0.762	0.730	0.731	0.781	0.731	0.740	0.771

Table 5. Results of the learning resource prerequisite inferring on the MOOCs Dataset

Metrics	Precision			Recall			F1-score		
	20	30	40	20	30	40	20	30	40
Training data(%)	20	30	40	20	30	40	20	30	40
CGL	0.631	0.661	0.672	0.612	0.604	0.633	0.625	0.631	0.652
PL-LDA	0.507	0.524	0.567	0.582	0.611	0.604	0.542	0.564	0.585
CANE	0.645	0.662	0.674	0.663	0.684	0.722	0.654	0.673	0.697
CPNE	0.752	0.774	0.791	0.727	0.764	0.780	0.739	0.769	0.785

6 Conclusion

In this paper, we propose an attention-based neural network model called CPRI to infer the concept prerequisites. CPRI extracts deep semantic features from concept texts and fuses them with the artificial features constructed based on the Wikipedia corpus. The fused features are input into a fully connected network to get the result of the relationship between concepts. Then, based on the obtained concept prerequisites, we design CPNE, a multi-role network embedding algorithm to learn the vector representation of each concept as the prerequisite and subsequent roles. The weights of the concepts contained in learning resources are calculated by TF-IDF, and the vectors of learning resources are combined by the weighted sum of concept vectors. Given a pair of learning resources, their vectors are passed to a Random Forest classifier to obtain the final result whether the prerequisite relationship exists. By conducting extensive experiments on two benchmark datasets, we compare the performance of CPRI and CPNE respectively. The experimental results show that our methods outperform the baselines with a significant improvement.

Acknowledgement. This work was supported by the National Key R&D Program of China under Grant 2018YFB0204100, the National Natural Science Foundation of China under Grants U1911201, 61802452, 62072486, Guangdong Special Support Program under Grant 2017TX04X148, and the project “PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (LZC0019)”.


References

1. Bromley, J., Guyon, I., Lecun, Y., Säckinger, E., Shah, R.: Signature verification using a siamese time delay neural network. In: Advances in Neural Information Processing Systems 6, 7th NIPS Conference, Denver, Colorado, USA, 1993 (1993)
2. Liang, C., Wu, Z., Huang, W., Giles, C.L.: Measuring prerequisite relations among concepts. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1668–1674 (2015)
3. Liang, C., Ye, J., Wang, S., Pursel, B., Giles, C.L.: Investigating active learning for concept prerequisite learning. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)

4. Liang, C., Ye, J., Wu, Z., Pursel, B., Giles, C.L.: Recovering concept prerequisite relations from university course dependencies. In: Thirty-First AAAI Conference on Artificial Intelligence (2017)
5. Lu, W., Zhou, Y., Yu, J., Jia, C.: Concept extraction and prerequisite relation learning from educational data. In: Proceedings of the AAAI Conference Artificial Intelligence, **33**, 9678–9685 (2019)
6. Luo, Z., Wang, Z., Wu, D., Hei, X., Du, Y.: Automatic generation and assessment of student assignments for parallel programming learning. In: Shen, H., Sang, Y. (eds.) Parallel Architectures, Algorithms and Programming - 10th International Symposium, PAAP 2019, Guangzhou, China, December 12–14, 2019, Revised Selected Papers. Communications in Computer and Information Science, vol. 1163, pp. 184–194. Springer (2019). https://doi.org/10.1007/978-981-15-2767-8_18
7. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
8. Nallapati, R.M., Ahmed, A., Xing, E.P., Cohen, W.W.: Joint latent topic models for text and citations. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 542–550. ACM (2008)
9. Pan, L., Li, C., Li, J., Tang, J.: Prerequisite relation learning for concepts in moocs. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers), pp. 1447–1456 (2017)
10. Roy, S., Madhyastha, M., Lawrence, S., Rajan, V.: Inferring concept prerequisite relations from online educational resources. In: Proceedings of the AAAI Conference Artificial Intelligence, **33**, 9589–9594 (2019)
11. Talukdar, P.P., Cohen, W.W.: Crowdsourced comprehension: predicting prerequisite structure in wikipedia. In: Proceedings of the Seventh Workshop on Building Educational Applications Using NLP. pp. 307–315. Association for Computational Linguistics (2012)
12. Tu, C., Liu, H., Liu, Z., Sun, M.: Cane: context-aware network embedding for relation modeling. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (vol. 1: Long Papers), pp. 1722–1731 (2017)
13. Vuong, A., Nixon, T., Towle, B.: A method for finding prerequisites within a curriculum. In: EDM, pp. 211–216 (2011)
14. Wang, S., Ororbial, A., Wu, Z., Williams, K., Liang, C., Pursel, B., Giles, C.L.: Using prerequisites to extract concept maps from textbooks. In: Proceedings of the 25th ACM International Conference on Information and Knowledge Management, pp. 317–326. ACM (2016)
15. Wang, Z., Wu, D., Luo, Z., Du, Y.: Building a lightweight container-based experimental platform for HPC education. In: Shen, H., Sang, Y. (eds.) Parallel Architectures, Algorithms and Programming - 10th International Symposium, PAAP 2019, Guangzhou, China, December 12–14, 2019, Revised Selected Papers. Communications in Computer and Information Science, vol. 1163, pp. 175–183. Springer (2019). https://doi.org/10.1007/978-981-15-2767-8_17
16. Yang, Y., Liu, H., Carbonell, J., Ma, W.: Concept graph learning from educational data. In: Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, pp. 159–168. ACM (2015)



Research on Bank Knowledge Transaction Coverage Model Based on Innovation Capacity Analysis

Ming Zhu, Zhenyu Wang^(✉), Xiangyang Feng, Pengyu Wan, Wenpei Shao, and Ran Tao

School of Computer Science and Technology, Donghua University, Shanghai, China
2181742@mail.dhu.edu.cn

Abstract. The asymmetry of bank-enterprise information makes banks unable to ensure the comprehensiveness and authenticity of corporate data. Therefore, the innovation ability of the enterprise cannot be accurately assessed. The data of each bank is similar, but because each bank has a different understanding of these data, different knowledge will be formed. In this paper, the idea of internal knowledge transaction between banks is proposed. We propose The bank Knowledge Transaction Coverage Model to ensure the success of the knowledge transaction. The model includes three stages: knowledge encoding, knowledge coverage, and optimal selection. In this model, transaction bank set is selected to obtain the maximum knowledge coverage with the lowest cost. Since the number of participating banks is determined, in order to ensure that the model can be solved in polynomial time, the KCSA-GA algorithm is proposed. What's more, the multi-objective optimization is transformed into minimization fitness function. Theoretical analysis and results demonstrate that compared with GA algorithm, KCSA-GA algorithm is superior in the distribution of fitness function and convergence in each iteration. After the knowledge transaction, the effectiveness of the model is verified by the innovation ability ranking comparing with the expert ranking and the Kendall rank correlation coefficient τ is 91.74%.

Keywords: Accuracy · Completeness · Bank Knowledge Transaction Coverage Model · KCSA-GA algorithm

1 Introduction

In order to accurately evaluate the innovation capabilities of enterprises, many scholars have done relevant research.

Jamil et al. [1] proposed a RESTful API-based trust rule library BRB expert system to evaluate the innovation ability of the enterprise. This method provides

This research is supported by the key laboratory of embedded system and service computing ministry of education (Tongji University).

an abstraction layer in the user's domain. It increases the portability, usability and computing power of BRB. Wu et al. [2,3] proposed A fuzzy ANP method using interval type-2 fuzzy sets to evaluate the innovation ability of enterprises. Yake et al. [4] established a mathematical model by using Delphi method and fuzzy comprehensive evaluation method to objectively evaluate the enterprise's independent innovation capability.

To evaluate the innovation capability of an enterprise, banks need to convert enterprise data into knowledge. The process from data to knowledge is the process of knowledge reasoning. Knowledge reasoning can be divided into three categories [5]: rule-based reasoning, distributed representation-based reasoning and neural network-based reasoning.

There are differences in business models, fields, and regions of various companies, and some documents cannot be made public, making it difficult to obtain and encode knowledge. So this paper evaluates the innovation ability of Sci-tech Medical Enterprise.

It is believed that companies can be evaluated through bank transaction. There is little difference in data between banks. However, due to differences in the literacy, intuition, and cooperation level of employees in each bank, the knowledge internalized from these data will also be different. These differences make knowledge trading possible. This is the key to a bank's uniqueness. These differences make knowledge transaction possible.

In this paper, each bank determines the set of transaction banks through the knowledge transaction model, which uses the smallest transaction cost to form a complete set of knowledge. In this way, banks will have a more comprehensive understanding of enterprises, which will contribute to reducing information asymmetry, improving the accuracy of evaluation.

The significant contributions of this paper are as follows:

- Each bank transforms the information of enterprises into different knowledge, in which explicit knowledge is expressed by knowledge weight, and tacit knowledge is expressed by knowledge coding degree.
- Construct knowledge transaction coverage model to select transaction banks, make transaction cost as small as possible while obtaining maximum knowledge coverage, and express the model with mathematical definition and formula.
- Improve genetic the algorithm to solve this model, convert the problem of multi-objective optimization into minimizing fitness function, redesign the gene coding and add constraint conditions, so that it will not generate invalid genes during crossover and mutation, thereby improving the algorithm performance.

2 Related Work

The knowledge creation cycle is divided into four parts: Socialization, Externalization, Combination, and Internalization. This is the SECI model [12-14], whether it is learning, growth or knowledge of human, it must be completed

in the context of social interaction. The reasoning of knowledge is the process of internalization, and the sharing of knowledge is the process of combination. There is little difference between corporate financial statements and corporate data obtained on the Internet. However, due to differences in the literacy, intuition, and cooperation among bank employees, the knowledge internalized from these data will also vary [18]. These differences make knowledge transaction possible. This is the key to a bank's uniqueness. If the bank trade this knowledge instead of data, the participating banks will understand the status of the enterprise from more dimensions and make the evaluation more accurate.

Knowledge transaction is considered from the following three aspects: knowledge encoding, knowledge sharing and model design. Many scholars have done related researches.

In aspect of knowledge encoding, knowledge includes explicit knowledge and tacit knowledge from the perspective of knowledge transaction. Zhu et al. [6] encoded the innovation ability of Sci-tech Medical Enterprise, analyzed the enterprise's patents, and reflected the innovation ability of enterprises from the breadth and depth of patents. Banerjee et al. [7] used rough set-theoretic concepts to encode knowledge and used the dependency factors to encode initial weight.

In aspect of knowledge sharing, many scholars have done related research. Tabatabaei et al. [8] proposed a nonlinear Bi-Level Programming (BLP) model to analyze the knowledge sharing behavior of academics, which considered the sharing of tacit knowledge and models of publishing coding knowledge. Sharing tacit knowledge is a time-consuming and hard task, it can support explicit knowledge sharing [9].

In this paper, a model of knowledge transaction coverage between banks is constructed to share corporate knowledge. In order to solve the model in polynomial time, the KCSA-GA algorithm is proposed where the encoding, crossover and mutation operations are optimized. It provides the completeness and authenticity of corporate data to evaluate corporate innovation capacities in all aspects and reduce loan risks.

3 Problem Statement and Design Goals

In this paper, in order to carry out the knowledge transaction (KT) successfully, a knowledge coverage model is built which includes three stages: knowledge encoding (KE), knowledge coverage (KC), and optimal selection (OS), that is, $KT = KE + KC + OS$.

Knowledge sharing [10] is defined as the process of communicating explicit or tacit knowledge to others. The knowledge to be shared must be explicit knowledge. Therefore, the tacit knowledge [11] must be encoded and expressed in strict logic before knowledge sharing. Knowledge encoding is the process of transforming the tacit knowledge (TK) into a symbol system (SS) that is accepted and identifiable by all banks. Let the encoding function be F , there is: $SS = F(TK)$.

After knowledge encoding, in order to obtain the required knowledge through knowledge transactions, the target bank needs to calculate the set of banks participating in the knowledge transaction.

At the KC stage, the knowledge provided by the banks involved in knowledge transactions should try to meet the knowledge requirements proposed by the target bank. However, the transaction requires costs. Therefore, we select the transaction bank set to obtain the maximum knowledge coverage and the lowest transaction cost.

At the OS stage, a knowledge coverage selection algorithm is designed to select the approximate optimal solution of the transaction bank set in polynomial time.

Then the entire process of knowledge discovery is shown as Fig. 1. Transaction banks encode tacit knowledge and combine it with explicit knowledge to form a knowledge set of transaction bank. With knowledge management and mode conversion, the knowledge can be understood by the target bank. Knowledge coverage is then used to select a subset to form the set of knowledge needed by the target bank. In this process, algorithms are designed to obtain the maximum knowledge coverage at the least cost. Finally, the target bank will store the shared knowledge.

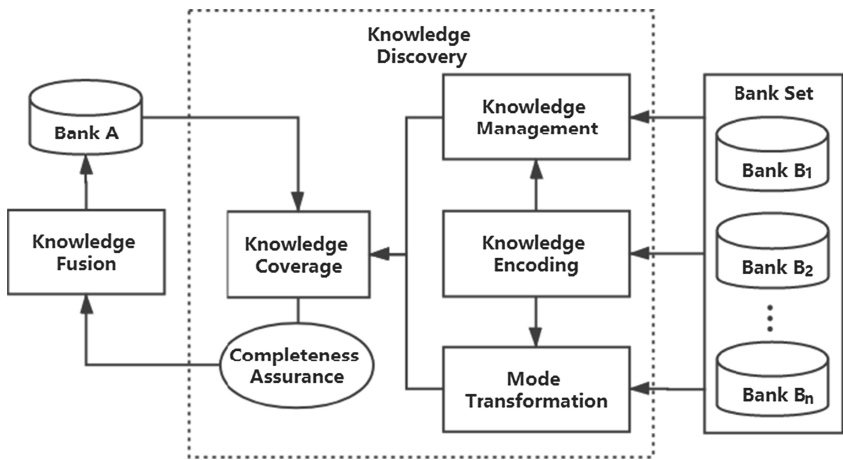


Fig. 1. Framework of knowledge transaction

4 Key Technologies and Models

4.1 Knowledge Encoding

Tacit knowledge is expressed in strict logical form through knowledge encoding. The difficulty with knowledge encoding is that it can hardly be represented in a

discrete form. It is believed that both the coding and abstraction of knowledge [15] can be regarded as a process of removing redundancy and reducing disturbing data to make the knowledge and expression clearer. However, it is believed that not all problems can be solved in strict logical form [11]. So, in this paper, for the tacit knowledge that is not easy to express, we use the degree of knowledge encoding to reflect the amount of knowledge contained in the encoded knowledge.

The followings are some assumptions about tacit knowledge encoding.

Assumption 1: For a certain knowledge required by the target bank, if a transaction bank has tacit knowledge related to it, this tacit knowledge will satisfy all the needs of the target bank for this knowledge. The purpose of this assumption is to directly express the satisfaction degree of the target bank with this knowledge through the degree of knowledge encoding.

Assumption 2: The tacit knowledge of a bank and its encoded knowledge can be quantified by information entropy.

Information entropy represents the uncertainty of which the macroscopic state is for an observer. Let the probability corresponding to each microscopic state be p_1, p_2, \dots, p_n and the formula of information entropy is: $H = E[-\log p_i] = - \sum_{i=0}^n p_i \log p_i$.

Let the information entropy of a tacit knowledge owned by the bank be H , and its information entropy after encoding is H' , then the degree of encoding of this knowledge w is: $w = \frac{H'}{H}$.

It should be noted that the tacit knowledge of banks exists in the brains of individual members. At the same time, the tacit understanding, coordination, and working skills of relevant organizations are also very important tacit knowledge resource, which are scarce, valuable, ambiguous, and difficult to imitate, thus becoming an important source of competitive advantages for banks.

4.2 Definition of Knowledge Transaction Coverage Model

The Knowledge Transaction Coverage Model can be used to represent how to use the minimum cost to maximize the knowledge coverage through a certain number of transaction bank knowledge sets. The mathematical definition is defined as follows:

Definition 1: An information system is a pair $S = (U, Z)$, where,

- 1) U is a non-empty finite set of objects;
- 2) Z is a non-empty finite set of attributes;
- 3) for every $z \in Z$, there is a mapping $a: U \rightarrow V_u(z)$, where $V_U(z)$ is called the value set of z .

If $V_U(z)$ contains a null value for at least one attribute $z \in Z$ then S is called an incomplete information system otherwise it is called a complete information system [16]. From now on, we will denote the null value by \star .

Definition 2: Target bank A. The target bank is a bank that needs to ensure the completeness and accuracy of its knowledge base through knowledge transactions. The set of knowledge owned by A is K_A and the set of knowledge that the target bank needs to supply through the transaction is $U - K_A$.

Definition 3: Set of transaction bank types B. The number of transaction banks is n_B , the transaction bank set can be defined as:

$$B = B_1, B_2, \dots, B_{n_B} \tag{1}$$

$$|B| = n_B$$

Definition 4: Participating transaction banks B_i . B_i is a knowledge set of a bank that may have a knowledge transaction with the target bank, $B_i \in B$. For each transaction bank type $B_i (1 \leq i \leq n_T)$, its knowledge understanding, knowledge weight, and knowledge pricing are different, so a triple is used to define it:

$$B_i = \langle S_i, W_i, p_i \rangle, i \in [1, n_T] \tag{2}$$

where p_i is the price of knowledge, and its type is float. The intersection of the knowledge set owned by B_i bank and the knowledge set required by bank A is recorded as S_i , $S_i = (U - K_A, T)$ is an incomplete information system, $T \subseteq Z$.

Definition 5: Weight matrix of knowledge W_i . Let $W = (W_i)$ be a set of matrices. For tacit knowledge, it represents encoding degree of different banks. For explicit knowledge, is the degree of expression of the knowledge [17].

Definition 6: In order to make it easy for banks to express their knowledge, we introduce the knowledge ownership matrix $C = (C_i)$ instead of S_i , C_i is a 0-1 matrix with $n_S C$ columns and $n_S L$ lines, where

$$C_k = (C_{kij}) = \begin{cases} 0, & V_{U-KA_i}(P_j) \text{ is } \star \\ 1, & \text{otherwise} \end{cases} \tag{3}$$

Definition 7: Selection vector \mathbf{x} : \mathbf{x} records the banks participating in the transaction.

Assumption 3: The degree of tacit knowledge encoded by the transaction banks can be superimposed, but if the degree of encoding after superimposition is greater than 100%, it is calculated as 100%, which means that the tacit knowledge has been fully expressed.

Definition of The Knowledge Transaction Coverage Model is as follows:

$$\begin{aligned}
 & \text{Set: } y_{kij} = w_{kij} \times c_{kij} \\
 y'_{kij} = & \begin{cases} 0, & V_{U-KA_i}(P_j) \text{ is } \star \\ 1, & \text{otherwise} \end{cases} \quad k \in [1, n_B], i \in [1, n_{SC}], j \in [1, n_{SL}] \\
 & \text{maximize } BK = \sum_{k=1}^{n_B} \left(\sum_{i=1}^{n_{SC}} \sum_{j=1}^{n_{SL}} y'_{kij} \right) \times x_k \\
 & \text{minimize } BP = \sum_{i=1}^{n_T} p_i x_i \\
 & \text{s.t. } x_i = 0 \text{ or } 1 \quad i \in [1, n_k] \\
 & c_{kij} = 0 \text{ or } 1 \quad k \in [1, n_b], i \in [1, n_{SC}], j \in [1, n_{SL}] \\
 & 0 \leq w_{kij} \leq 1 \quad k \in [1, n_b], i \in [1, n_{SC}], j \in [1, n_{SL}]
 \end{aligned} \tag{4}$$

4.3 KCSA-GA Algorithm Design

In this section, KCSA-GA (Knowledge Coverage Select Algorithm based on Genetic Algorithm) is proposed to solve the knowledge coverage problem. The following is the implementation process of KCSA-GA.

Gene Encoding. The genes are non-binary coding. The length of each chromosome is n_k , and the gene is $g = [g_1, g_2, \dots, g_{n_k}]$, where the value of each gene is $g_i \in [1, n_b]$, indicating that transaction bank with the number gi is selected for the i -th time. Correspondingly, in order to prevent the transaction bank from being selected repeatedly, it is stipulated that the value of genes should follow a

strictly monotonous increasing order, which is $\begin{cases} g_i = 0, & i = 0 \\ g_{i-1} \leq g_i, & 1 \leq i \leq n_k + 1 \\ g_i = n_b + 1, & i = n_k + 1 \end{cases}$

Among them, g_0 and g_{n_k+1} are auxiliary genes which are designed to facilitate description and programming.

Fitness Function. The initial problem is a multi-objective optimization problem. In this paper, it is transformed into a fitness function, and the solution of it is transformed into a process that minimizes the fitness function:

$$f = \sqrt{(BP - \widehat{BP})^2 + \left(\frac{BK - \widehat{BK}}{\widehat{KA} - \widehat{KA}}\right)^2}.$$

Among them, \widehat{BP} and \widehat{BK} are the price and amount of knowledge in the ideal state of knowledge transaction. In the ideal state, the price of knowledge is its minimum price, and the amount of knowledge is the largest amount of knowledge. $\widehat{BP} = \min(p_i | i \in [1, n_B]) \times n_k$, $\widehat{BK} = n_{SC} \times n_{SL} \times n_K \cdot \widehat{KA}$ is the minimum amount of knowledge of banks, $\widehat{KA} = \min\left(\sum_{i=1}^{n_{SC}} \sum_{j=1}^{n_{SL}} c_{kij} \times w_{kij} | i \in [1, n_B]\right)$.

Crossover. Because adjacent genes have constraints, KCSA-GA improves on the crossover operation. For chromosomes ch_1 and ch_2 , and their crossover probability P_c , the crossover operation point $point_1$ of ch_1 is generated by random, and the first point $point_2$ on ch_2 is founded that meets the constraints, and then the values of these two points are exchanged.

Mutation. In KCSA-GA, each gene mutates with a probability of P_m . In the process of gene mutation operation, due to constraints, the upper and lower bounds of the mutation are determined.

Time Complexity of KCSA-GA Algorithm. We set that the population of the algorithm is $popsize$, the crossover probability is P_c , the average number of mutations is P_m , the complexity of each calculation of the fitness function is C_f , and the number of iterations is $iter$. The time complexity of KCSA-GA algorithm is $O(popsize^3 \times P_c \times P_m \times C_f \times iter)$.

KCSA-GA. The pseudo code of KCSA-GA is as follows (Table 1):

Table 1. The pseudo code of KCSA-GA

Algorithm 1. KCSA-GA

input: func, size_population, max_iter, size_chrom,P_c,P_m
output: best selection scheme of Knowledge Transaction Coverage Model
begin
 initial: chrom ← generate_chrom(size_population , size_chrom)
 for i = 0, 1, ⋯ , max_iter do
 chrom ← ranking(chrom)
 chrom ← selection(chrom)
 chrom ← KCSA_crossover(chrom,P_c)
 chrom ← KCSA_mutation(chrom, P_m)
 record the best ones
 end
 bestScheme ← global best selection scheme
end

5 Experiments and Results

5.1 Experimental Data

This paper uses the data of 218 Sci-tech Medical Enterprise from 2008 to 2019. According to the scientific and technological medical enterprise evaluation system [17], 23 innovation capability indicators were established. According to data

released by the China Banking and Insurance Regulatory Commission, as of the end of 2018, there were 4,056 banking institutions in China. So the number of transaction banks is 4056. The missing knowledge of each bank is normally distributed, and the number of missing is *sampleNo*.

For explicit knowledge, we input the textual knowledge of the company and use jieba word segmentation to divide them into words. Every bank has different understanding of corporate knowledge, but we do not have the real knowledge of all banks. Random characters are added to simulate noise. Then we use word2vec to vectorize these words, and explicit knowledge is represented by the average of all word vectors. The degree of expression of explicit knowledge is expressed by the cosine similarity between the knowledge vector with noise and the knowledge vector without noise.

For the price of knowledge, stochastic number is used to simulate the price. For each bank, two pieces of tacit knowledge are encoded and stochastic number is used to simulate the encoding degree of tacit knowledge. A sample table of bank knowledge is shown as follows (Table 2):

Table 2. A sample table of bank knowledge

Sci-tech Medical Enterprise	T_1	T_2	T_3	\dots	T_{23}	T_{24}	T_{25}	Price
E_1	0.56	★	★	\dots	0.84	1	1	0.87
E_2	0.83	0.26	0.14	\dots	★	1	1	0.35
E_3	0.25	★	0.07	\dots	★	1	1	0.52
\dots	\dots	\dots	\dots	\dots	\dots	\dots	\dots	\dots

In order to verify the efficiency and effectiveness of the KCSA-GA algorithm, in this paper, the KCSA-GA algorithm and the GA algorithm are used to compare the distribution of fitness function and the convergence effects. The parameters are as follows (Table 3):

Table 3. Parameters of KCSA-GA and GA

Symbol	Description	Value
<i>sampleNo</i>	Amount of knowledge owned by each bank	2500
n_B	Number of transaction banks	4056
n_T	Number of banks involved in the transaction	50
n_{SC}	Number of columns of the ownership matrix	218
n_{SL}	Number of rows of the ownership matrix	25
<i>sizepop</i>	Population in the algorithm	50
p_c	The probability of crossover operation	0.8
p_m	The probability of mutation operation	0.01

5.2 Results

While using the above parameters, both algorithms generally converge when iterating for about 30 times. In order to demonstrate better, the first 40 iterations are selected in this experiment. The GA algorithm does not constrain the range of genes, so genes that do not satisfy the constraints may be generated after crossover and mutation operations. At this time, $f = \infty$, but for better demonstration, $f = 45$. The followings are the comparison results of the KCSA-GA algorithm and the GA algorithm (Figs. 2 and 3).

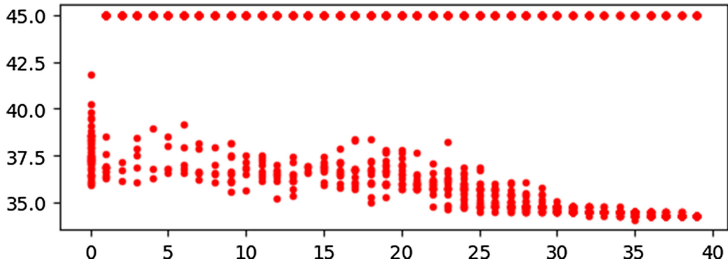


Fig. 2. The distribution of f in each iteration of GA algorithm

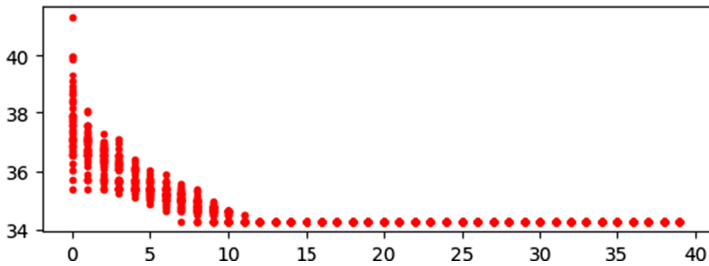


Fig. 3. The distribution of f in each iteration of KCSA-GA algorithm

Both GA algorithm and KCSA-GA algorithm will eventually converge to minimum. But because of the constraints between the genes of KCSA-GA algorithm, no invalid genes will be generated. Therefore, it is superior to GA algorithm in both the distribution of fitness function and the convergence in the iteration.

Table 4. An enterprise evaluation after knowledge transaction in different years

Year	Bank number							Score
2008	24	50	201	246	379	...	3920	80
2009	35	97	158	326	468	...	3847	83
2010	53	107	257	485	630	...	3950	85
...
2019	21	85	164	288	529	...	4002	84



Fig. 4. The convergence of f when each bank has a different average amount of knowledge

From Table 4, it is shown that the banks the target bank chooses to trade after knowledge transactions in different years, and the scores of the company’s innovation capabilities.

From Fig. 4, we conclude that the fitness function f decreases with the increase of *sampleNo* when each bank has a different average amount of knowledge.

5.3 Verification

In this paper, in order to verify the accuracy of this experiment, we use fuzzy judgment to score the scientific and technological innovation enterprises. Kendall rank correlation coefficient is used to verify the effect of this model through the ranking of innovation ability after knowledge trading comparing with the expert ranking [17]. As a result, the Kendall rank correlation coefficient $\tau = 91.74\%$.

6 Discussion

In this paper, knowledge is shared between banks through transactions, so as to ensure that banks have completeness and accuracy of corporate data. This is the data support for banks to accurately evaluate the innovation capacities of enterprises. The innovations of this paper are as follows. **firstly**, the tacit knowledge and explicit knowledge owned by each bank is considered. We use the degree of coding to express tacit knowledge and use degree of expression to represent explicit knowledge. **Secondly**, we build the Knowledge Transaction Coverage Model and explain how knowledge is traded in order to achieve the maximum knowledge coverage with the minimum cost. Then the model is defined by a mathematical formula. **Thirdly**, for the model, we design KCSA-GA algorithm. The advantage of this algorithm is that it solves the multi-objective optimization problem by designing the fitness function. We optimize the gene coding according to the characteristic of the participating transaction banks and design the constraint conditions of the gene. So no invalid genes will be generated during crossover and mutation. The algorithm we proposed is better than the traditional genetic algorithm.

The shortcomings of this paper are: **firstly**, while the multi-objective optimization is transformed into the fitness function, the importance of each objective is not considered. **Secondly**, the smallest unit of knowledge transaction of a bank is the entire knowledge of the bank, and no consideration is given to purchasing a certain piece of knowledge in a bank's knowledge base.

References

1. Jamil, M.N., Hossain, M.S., ul Islam, R., Andersson, K.: A belief rule based expert system for evaluating technological innovation capability of high-tech firms under uncertainty. In: 2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR), Spokane, WA, USA, pp. 330–335 (2019). <https://doi.org/10.1109/ICIEV.2019.8858550>
2. Wu, T., Liu, X.: An interval type-2 fuzzy ANP approach to evaluate enterprise technological innovation ability. *Kybernetes* **45**(9), 1486–1500 (2016)
3. Wu, T., Liu, X.W., Liu, S.L.: A fuzzy ANP with interval type-2 fuzzy sets approach to evaluate enterprise technological innovation ability. In: 2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE). IEEE (2015)
4. Yake, C., Yongsheng, S.: The study of fuzzy comprehensive evaluation on enterprise's independent innovation ability (ID:K-016). In: The Proceedings of the 14th International Conference on Industrial Engineering and Engineering Management (Volume B) (2007)
5. Chen, X., Jia, S., Xiang, Y.: A review: knowledge reasoning over knowledge graph. *Expert Syst. Appl.* **141**, 112948.1–112948.21 (2020)
6. Zhu, M., Wan, P., Feng, X., Wang, Z., Shao, W.: Research on network awareness of enterprise evaluation system indicators. In: IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), Madrid, Spain, pp. 311–315 (2020). <https://doi.org/10.1109/COMPSAC48688.2020.0-228>

7. Banerjee, M., Mitra, S., Pal, S.K.: Rough fuzzy MLP: knowledge encoding and classification. *IEEE Trans. Neural Netw.* **9**(6), 1203–1216 (1998)
8. Tabatabaei, M., Afrazeh, A., Seifi, A.: A game theoretic analysis of knowledge sharing behavior of academics: bi-level programming application. *Comput. Ind. Eng.* **131**(MAY), 13–27 (2019)
9. Charband, Y., Jafari Navimipour, N.: Knowledge sharing mechanisms in the education: a systematic review of the state of the art literature and recommendations for future research. *Kybernetes* **47**, 1456–1490 (2018). <https://doi.org/10.1108/K-06-2017-0227>
10. Becerra-Fernandez, I., Sabherwal, R.: *Knowledge Management: Systems and Processes*. M.E. Sharpe, New York (2010)
11. Hui, K.: The codify of tacit knowledge. *Stud. Dialect. Nat.* **1**, 6 (2005)
12. Nonaka, I., Konno, N.: The concept of “Ba”: building a foundation for knowledge creation. *Calif. Manag. Rev.* **40**(3), 40–54 (1998)
13. Nonaka, I., Toyama, R., Konno, N.: SECI, Ba and leadership: a unified model of dynamic knowledge creation. *Long Range Plan.* **33**(1), 5–34 (2000)
14. Nonaka, I., Toyama, R.: The knowledge-creating theory revisited: knowledge creation as a synthesizing process. *Knowl. Manag. Res. Pract.* **1**(1), 2–10 (2003)
15. Boisot, M.H.: *Knowledge Assets: Securing Competitive Advantage in the Information Economy*. OUP, Oxford (1998)
16. Kryszkiewicz, M.: Rules in incomplete information systems. *Inf. Sci.* **113**(3–4), 271–292 (1999)
17. Shao, W., Feng, X., Zhu, M., Tao, R., Lv, Y., Shi, Y.: Fuzzy evaluation system for innovation ability of science and technology enterprises (KMO 2020 accepted)
18. Wipawayangkool, K., Teng, J.T.C.: Profiling knowledge workers’ knowledge sharing behavior via knowledge internalization. *Knowl. Manag. Res. Pract.* **17**, 1–13 (2018)



Deep Deterministic Policy Gradient Based Resource Allocation in Internet of Vehicles

Zhuo Ma^{1(✉)}, Xin Chen¹, Teng Ma², and Ying Chen¹

¹ School of Computer Science, Beijing Information Science and Technology University, Beijing 100101, China

{mazhuo, chenxin, chenying}@bistu.edu.cn

² School of Automation, Beijing Information Science and Technology University, Beijing 100101, China

2019020223@bistu.edu.cn

Abstract. The development of sensors and wireless communication technology has greatly promoted the development of the Internet of Vehicles (IoV). In Vehicle to Everything (V2X) communication, various types of services continue to emerge. Due to the different Quality of Service (QoS) requirements of the services, we assuming that the total bandwidth of uplink is equally divided into several orthogonal sub-band, each Vehicle-to-Infrastructure (V2I) link is pre-allocated with a sub-band, and Vehicle to Vehicle (V2V) communication can transmit information through spectrum multiplexing. We model the spectrum allocation and power selection as a Markov Decision Process (MDP). For the case of continuous action space, Deep Deterministic Policy Gradient (DDPG) is a promising method. We propose a DDPG based Spectrum and Power Allocation (DSPA) algorithm. The simulation results show that the proposed algorithm can effectively learn the appropriate resources allocation strategy from the environment and meet the QoS requirements of different services.

Keywords: Internet of Vehicles · Resource allocation · Quality of service · DDPG · Markov decision process

1 Introduction

With the vigorous development of sensor and wireless vehicle communication technology, vehicle can obtain environmental data around itself through sensors such as cameras and radars [1], and wireless vehicle communication technology allows vehicle to communicate with surrounding devices, expanding the vehicles perception range [2]. Therefore, the concept of the Internet of Vehicles (IoV) came into being and attracted widespread attention from the academic community.

The IoV means that vehicles communicate with everything, i.e., Vehicle to Everything (V2X), including Vehicle-to-Infrastructure (V2I), Vehicle to Vehicle (V2V), Vehicle-to-Person (V2P), etc. Generally speaking, different types of

communication methods have different Quality of Service (QoS) requirements. Specifically, V2I communication needs to guarantee greater throughput, while V2V communication and V2P communication have strict requirements on delay and reliability. However, Long Term Evolution (LTE) system focuses on the communication between vehicles and infrastructure, and seldom involves in other communication methods, so it's not easy to ensure the needs of multiple communication methods in IoV. With the development of wireless communication technology, the 5-th Generation Mobile Communication Technology (5G) could meet the QoS requirements of various services in V2X communication. 5G allows direct device-to-device (D2D) communication, and creates conditions for V2X communication [3]. Therefore, 5G IoV has broad development prospect. D2D communication shows great potential in 5G [4], so in this article, we consider the V2X communication method based on D2D communication, where V2V communication can reuse the spectrum of V2I communication to allocate resources reasonably, and meet the QoS requirements of different vehicle users.

In the D2D communication problem, a key issue is how to reduce the interference between the D2D link and the cellular link. In [5], the author proposed a new sharing paradigm, allowing devices to share resources without involving cellular networks, thereby reducing interference. In [6], the interference control and minimization of system energy consumption are expressed as a mixed non-integer linear program. However, in these methods, the equipment is mostly stationary or slowly moving, and do not consider QoS requirements too much. In [7], the author constructed a Markov Decision Process (MDP) for power control in the 5G ultra-dense network to meet the real-time needs of users. A spectrum sharing algorithm based on Lyapunov Optimization is proposed in [8], which maximize the energy efficiency of the system while ensuring the QoS requirements of different users. These methods take into account the guarantee of user QoS requirements. However, users QoS requirements are often diverse, and some requirements are often difficult to express directly into optimization problems. For this problem, Deep Reinforcement Learning (DRL) method is a promising method.

DRL as a paradigm of machine learning, has made remarkable achievements in playing games and other fields [9]. DRL method also has broad application prospects in the fields of communication and network [10]. In [11], Deep Q-Learning (DQN) algorithm is used to deal with flexible numerology problems in 5G heterogeneous networks. In [12], a multi-agent deep Q-learning method is used to allocate spectrum resources in IoV. However, the DQN algorithm can only be used to process discrete action spaces. For the continuous action, the Deep Deterministic Policy Gradient (DDPG) algorithm is needed. In [13], the author proposed a DDPG-based pre-caching method to cache entertainment information in IoV, the size of the file to be cached changes continuously.

In this article, we have studied the issues of spectrum allocation and power selection in vehicle communication network. The main contributions of the article are as follows:

- We consider the spectrum allocation and power selection problem in the IoV scenario. Each V2V communication reuses the spectrum of V2I link and selects the appropriate transmission power to meet the QoS requirements of V2X communication.
- We model the spectrum and power allocation problem as a MDP. Since the action space is continuous, we propose a DDPG-based Spectrum and Power Allocation (DSPA) algorithm to solve this problem.
- Simulation experiments show that the proposed DSPA algorithm can be effective in learning appropriate spectrum allocation and power selection methods during the environmental interaction process.

The structure of this article is as follows. The system model is introduced in Sect. 2. In Sect. 3, the MDP model is proposed and the optimization objective is given. The DSPA algorithm is proposed in Sect. 4. The simulation results are given in Sect. 5 and Section 6 summarizes the article.

2 System Model

In this section, we will introduce the model of the vehicle communication networks. Considering a vehicle communication scenario with a base station (BS), as shown in Fig 1. There are M vehicles belonging to cellular users (CUEs), denoted by $\mathbf{M} = \{1, 2, \dots, M\}$, which need to establish a high-rate uplink connection with the BS, i.e., V2I link. At the same time, there are N vehicle pairs, called DUEs, carry out vehicle-to-vehicle communication based on D2D communication to transmit important safety information, and denoted by $\mathbf{N} = \{1, 2, \dots, N\}$. Note that each vehicle can be both CUE and DUE at the same time. Therefore, each vehicle has two types of power for sending information, denoted as P^c and P^d .

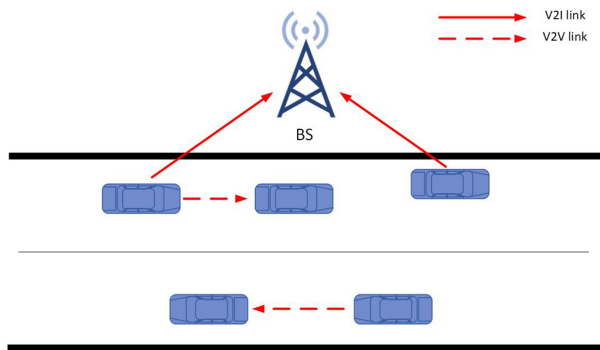


Fig. 1. An illustrative of V2X communication networks

Assuming that the total bandwidth of the uplink is W_{all} , consists of several equal orthogonal sub-bands. Each V2I link occupies a sub-band, that is, The

m -th CUE sends information through the m -th sub-band. Furthermore, since the uplink resource usage is relatively small and the base station has a certain anti-interference ability, we assume that each V2V communication can reuse an V2I link to improve the efficiency of spectrum utilization. Marking $\alpha_{n,m} \in \{0, 1\}$ as the symbol of spectrum reuse, that is, if the n -th V2V link reuse the spectrum of the m -th V2I link, $\alpha_{n,m} = 1$, otherwise, $\alpha_{n,m} = 0$. In order to avoid excessive interference, each DUE user can reuse the spectrum of one CUE user at most, and each CUE user can be reused by up to z DUE users. Consequently, $\sum_{m=1}^M \alpha_{n,m} \leq 1$ for all n , and $\sum_{n=1}^N \alpha_{n,m} \leq z$ for all m .

Thus, the Signal to Interference plus Noise Ratio (SINR) of the m -th V2I link, γ_m^c , can be expressed as:

$$\gamma_m^c = \frac{P_m^c g_m^c}{\sigma^2 + \sum_{n=1}^N \alpha_{n,m} P_n^d g_n^{d,c}}, \tag{1}$$

where g_m^c denotes the channel gain from the m -th CUE to the BS, σ^2 denotes the noise power, and $g_n^{d,c}$ is the interference gain of the n -th DUE reuse the m -th CUE's spectrum.

Similarly, the SINR of the n -th V2V link, γ_n^d , can be expressed as:

$$\gamma_n^d = \frac{P_n^d g_n^d}{\sigma^2 + I_n}, \tag{2}$$

where g_n^d denotes the channel gain from the n -th V2V communication over the m -th sub-band, and I_n is the interference power. The interference power is divided into two parts, the first is the interference from the m -th V2I link to the n -th V2V link, and the second is from the n' th ($n' \neq n$), which reuse the m -th V2I link, to the n V2V link. Thus, I_n can be expressed as:

$$I_n = P_m^c g_m^{c,d} + \sum_{n' \neq n}^N \alpha_{n',m} P_{n'}^d g_{n'}^{d,d}, \tag{3}$$

with $g_m^{c,d}$ and $g_{n'}^{d,d}$ represent the interference from the m -th V2I communication to the n V2V communication and the interference from the n' -th V2V communication to the n -th V2V communication over the same sub-band, respectively.

Hence, according to Shannon's theorem, the rate of the m -th CUE and n -th DUE can be expressed as:

$$r_m^c = W \log(1 + \gamma_m^c), \tag{4}$$

and

$$r_n^d = W \log(1 + \gamma_n^d), \tag{5}$$

where W is the bandwidth of each sub-channel.

Our goal is to select the appropriate spectrum and transmit power for V2V communication to meet its low latency and high reliability requirements, minimize the interference to other communication links, and maximize the sum rate

of V2I link. Therefore, we set the sum V2I rate as the optimization goal, and express the delay constraint as a reward function, giving a penalty when the constraint is violated.

3 Problem Formulation

The resource allocation problem mentioned above is dynamically changing, because services in IoV are continuously generated, and channel state information is also changing in real time. For this situation, traditional optimization methods are difficult to handle. A promising solution is to model the spectrum allocation and power selection problem as a MDP.

MDP consists of four parts $\{S, A, P, R\}$, where S denotes the set which include all possible states during time T , A is the action space, $P(s'; s, a)$ denotes the transition probability, and R represents the reward. T consists of several time slots of equal length in the time domain, and the length of each time slot is $|t|$. We treat the V2V link as an agent, at the beginning of each time slot t , the agent observes the environment state s_t and executes an action a_t . After a time slot, the system state becomes s_{t+1} and gets reward r_t .

3.1 System State Space

The state space is represented as $S = \{g_t^d, I_{t-1}, D_t, O_t\}$. Among them, g_t^d is the instant channel state information corresponding to the V2V link, I_{t-1} is the interference power in the previous time slot, which can be calculated by formula (3). D_t and O_t represents the size of the task to be transferred and the remaining time allowed for transmission, respectively.

3.2 Action Space

Action space is the set of actions selected for the V2V link, including the spectrum reuse factor $\alpha_{n,m}$ and the transmit power P^d . The spectrum allocation strategy can be obtained by determining the value of $\alpha_{n,m}$. Realistically, we set the transmit power to a continuous value from 0 to P_{max} , and $p^d = 0$ means that in the t -th time slot, no information is sent on the V2V link.

3.3 System Reward

We design the reward function into three parts. The first two parts represent the sum V2I rate and sum V2V rate, which can be calculated by formula (4) and formula (5), respectively. Furthermore, the reward function should also include constraints on the amount of tasks to be transmitted, so we design the transmission time as a negative reward, which represents a penalty. The reward function is expressed as:

$$r_t = \omega_1 \sum_{m=1}^M r_m^c + \omega_2 \sum_{n=1}^N r_n^d - \omega_3(O_0 - O_t), \quad (6)$$

where O_0 is the maximum allowed transmission time of V2V task, $O_0 - O_t$ denotes the transmission time, ω_1, ω_2 and ω_3 denotes the weights of the three parts, respectively. The first two parts of the reward are used to reduce the interference, and $O_0 - O_t$ represents the penalty. The longer the transmission time, the greater the penalty, prompting the agent to send the task as quickly as possible.

It is worth noting that high current rewards may not guarantee the long-term benefits of the system. So we must consider the current rewards and subsequent rewards. The goal is to maximize the expected cumulative discount reward. For our system, the expected cumulative discounted rewards can be expressed as:

$$R_t = \mathbb{E}[\sum_{i=0}^{\infty} \gamma^i r_{t+i}], \quad (7)$$

where $\gamma \in (0, 1)$ is the discount factor, affect how to make a trade-off between the latest reward and the future reward.

In summary, the optimization goal of this article is to maximize the system reward, i.e.,

$$\max_{\alpha_n, m, P^d} \sum_{t=0}^T R_t. \quad (8)$$

4 DDPG-Based Spectrum and Power Allocation

When designing dynamic resource allocation algorithms in IoV, Q-learning and deep Q-learning (DQN) are commonly used solutions. The update process of the DQN algorithm can be expressed as follows:

$$Q(s, a) = Q(s, a) + \alpha(r + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s, a)). \quad (9)$$

During the update process, the agent first completes the evaluation of the policy, and then improves the policy. In a limited action space, this update method is feasible, but if the action space is continuous, this calculation and selection cannot be performed. For the continuous action space, the DDPG algorithm is a promising method. So we tend to use the DDPG method to allocate spectrum and power for DUE users in this work. The DDPG algorithm was first proposed by David Silver et al. [14]. As shown in Fig. 2, DDPG adopts the form of actor-critic, include four nets, namely primary actor net, primary critic net, target actor net and target critic net. After training, DDPG can simulate real Q-tables without worrying about the explosion of dimensionality, and can generate deterministic strategies instead of random strategies.

The DSPA algorithm is shown in Algorithm 1. In the algorithm, a policy network with a parameter of θ^μ is used to represent the deterministic policy $a = \mu(s|\theta^\mu)$, the network receives the current state s and generates a definite action a . As the name suggests, this network also used to update the policy, corresponding to the actor in the actor-critic algorithm framework. The value

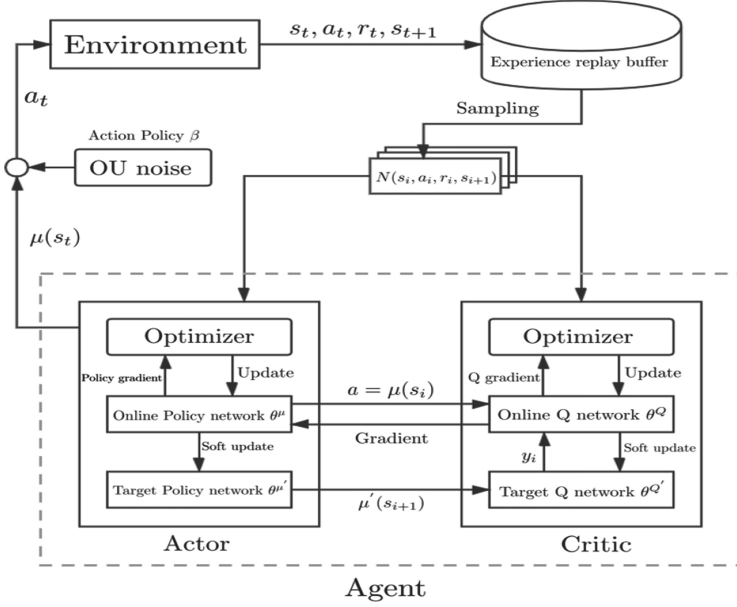


Fig. 2. DDPG algorithm framework diagram

network with parameter θ^Q is used to represent action value function $Q(s, a|\theta^a)$, which is used to solve the Bellman equation. The value function also corresponds to the critic in the actor-critic algorithm framework, used to provide gradient information. The DDPG algorithm does not only consider appropriate rewards, but also considers long-term discount rewards, i.e.,

$$J_\beta = \mathbb{E}_\mu[r_1 + \gamma r_2 + \dots + \gamma^n r_n], \tag{10}$$

where J_β is the objective function, and β is the behavior policy. Obviously, the policy that maximizes $J_\beta(\mu)$ and the optimal behavior policy μ^* are equivalent. μ^* can be expressed as:

$$\mu^* = \underset{\mu}{\operatorname{argmax}} J(\mu). \tag{11}$$

It has been proved in [14] that the gradient of $J_\beta(\mu)$ with respect to θ^μ and the expected gradient of $Q(s, a; \theta^Q)$ with respect to θ^μ are equal. Using the chain derivation rule to derive the objective function, the update process of the actor network is as follows:

$$\begin{aligned} \nabla_{\theta^\mu} J &\approx \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_{\theta^\mu} Q_\mu(s_t, \mu(s_t))] \\ &= \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_{\theta^\mu} Q(s, a; \theta^Q)|_{s=s_t, a=\mu(s_t; \theta^\mu)}]. \end{aligned} \tag{12}$$

Because the deterministic policy is $a = \mu(s; \theta^\mu)$, formula (12) can be rewritten as

$$\nabla_{\theta^\mu} J = \mathbb{E}_{s_t \sim \rho^\beta} [\nabla_a Q(s; \theta^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu} \mu(s_t; \theta^\mu)|_{s=s_t}]. \tag{13}$$

For formula (13), using the method of updating the value network in [12] to update the critic network, the gradient can be expressed as:

$$\nabla_{\theta^\mu} = \mathbb{E}_{s,a,r,s' \sim R}[(TargetQ - Q(s, a; \theta^Q)) \nabla_{\theta^Q} Q(s, a; \theta^Q)], \quad (14)$$

where

$$TargetQ = r + \nabla_{\theta^{Q'}} \gamma Q'(s', \mu(s'; \theta^{\mu'})). \quad (15)$$

The gradient descent algorithm is used to update the parameters in the network model. The process of training the value network is to find the optimal solution process of the parameter θ^Q in the value network.

Algorithm 1. DDPG-based spectrum and power allocation

- 1: Randomly initialize the weight parameter θ^Q of the primary critic network $Q(s, a; \theta^Q)$
 - 2: Randomly initialize the weight parameter θ^μ of the primary actor network $\mu(s; \theta^\mu)$
 - 3: Initialize the weight parameter $\theta^{Q'}$ of the target critic network $Q'(s, a; \theta^{Q'})$
 - 4: Initialize the weight parameter $\theta^{\mu'}$ of the target actor network $\mu'(s; \theta^{\mu'})$
 - 5: Initialize experience replay buffer R
 - 6: **for** *episode* = 1, J **do**
 - 7: Random initialization process E for action exploration
 - 8: Initialize the 5G heterogeneous vehicle communication network
 - 9: **for** *time* = 1, T **do**
 - 10: Calculate the action $a_t = \mu(s_t; \theta^\mu) + E_t$ at the current time step according to the current noisy policy
 - 11: Perform action a_t , record the reward r_t and new state s_{t+1}
 - 12: Store conversion experience (s_t, a_t, r_t, s_{t+1}) in experience replay buffer R
 - 13: Randomly sample N conversion experience samples (s_i, a_i, r_i, s_{i+1}) from the experience replay buffer R
 - 14: Calculate the target value y_i on $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}; \theta^{\mu'}); \theta^{Q'})$
 - 15: Update the primary critic network with the minimized loss function $L = \frac{1}{N} \sum_i (y_i - Q(s_i, a_i; \theta^Q))^2$
 - 16: Update the primary actor neural network by the sampled policy gradient ∇_{θ^μ}
 - 17: Update the target networks with:

$$\theta^{Q'} = \tau \theta^Q + (1 - \tau) \theta^{Q'}$$

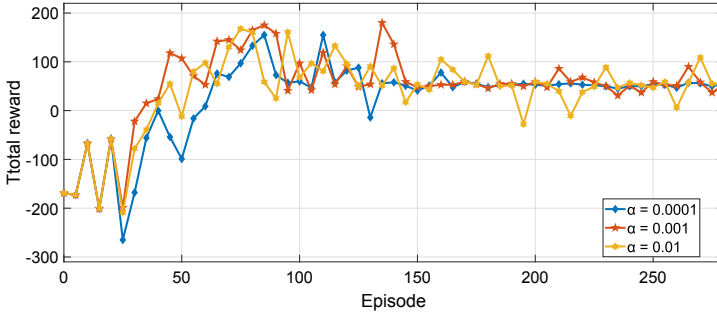
$$\theta^{\mu'} = \tau \theta^\mu + (1 - \tau) \theta^{\mu'}$$
 - 18: **end for**
 - 19: **end for**
-

5 Simulations Result and Discussion

In this section, we verify the performance of the proposed DSPA algorithm under different conditions. The relevant parameter settings of the simulation are shown in Table 1.

Table 1. Parameter setting in simulation and experiment

Parameter	Value
Carrier frequency	2 GHz
Bandwidth per channel	1.5 MHz
BS antenna height	25 m
BS antenna gain	8 dBi
Vehicle antenna height	1.5 m
Vehicle antenna gain	3 dBi
Vehicle speed	36 km/h
V2I transmission power	23 dBm
V2V maximum transmission power	23 dBm
Latency constraints for V2V links	100 ms
V2V maximum connection distance	150 m
Noise power σ^2	-114 dBm
$[\omega_1, \omega_2, \omega_3]$	[0.1, 0.9, 1]

**Fig. 3.** Comparisons of different learning rate α

In the proposed DSPA algorithm, the hyperparameter α has a huge impact on the convergence of results. If we set too large learning rate, a large amount of abnormal data will be generated, causing the results to diverge easily. If we set a very small learning rate, the convergence speed will be reduced and overfitting may occur. To compare the influence of learning rate on the convergence of the DSPA algorithm, We carried out three simulations, corresponding to three learning rates. Specifically, $\alpha = 0.01, 0.001$ and 0.0001 . The result is shown in Fig. 3, the abscissa represents the number of episodes, and the ordinate represents the reward. As can be seen from the figure, $\alpha = 0.01$ and $\alpha = 0.001$ will cause large flucence. When α is equal to 0.0001 , the convergence value will be relatively stable, and the convergence speed is not too slow. Therefore, in subsequent experiments, we will set the learning rate to 0.0001 , i.e., $\alpha = 0.0001$.

We compared the DQN-based algorithm with the proposed DSPA algorithm. In the simulation based on the DQN algorithm, we discretized the transmit power into 5 uniform values. The result is shown in Fig. 4. At the beginning of the episode, The DQN algorithm showed slightly better performance. But then, the performance of the DSPA algorithm began to exceed the DQN-based algorithm. Part of the reason is the network of the DSPA algorithm consists of actor network the critic network. Other is DSPA method can make a suitable choice from continuous action space choices, while the actions that can be selected by the DQN algorithm are limited.

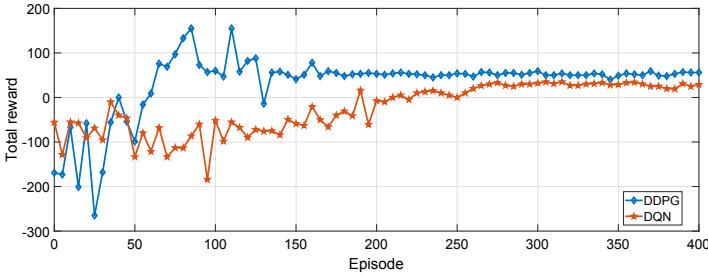


Fig. 4. Comparison of different algorithms

Figure 5 shows the performance comparison of DSPA algorithm, DQN algorithm and random algorithm based on the average sum rate. With the increasing number of vehicles, the rate of the three methods all show a downward trend. This is because as the number of vehicles increases, interference will increase. Among the three algorithms, the average sum rate of the DSPA method is the largest, and the performance of the random algorithm is the worst. This is because the random algorithm randomly selects the spectrum sub-band and transmit power by V2V communication, which will cause unimaginable interference.

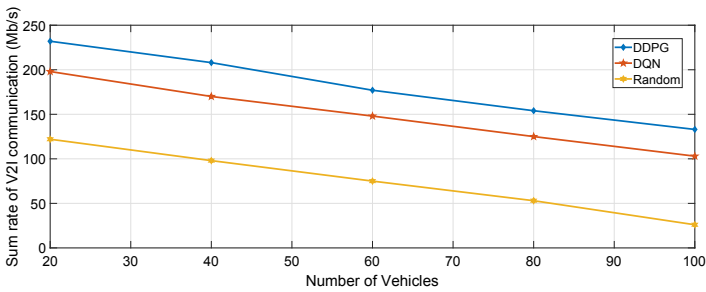


Fig. 5. Average sum-rate of V2I with different number of vehicles

Figure 6 shows the performance comparison of DSPA algorithm, DQN algorithm and random algorithm based on the probability of delivery. We assume that all data successfully sent within the specified time will be successfully delivered, while messages not sent over time will be discarded. As the number of vehicles increases, the probability of successful delivery of all vehicles gradually decreases. This is because the increase of vehicles will bring more interference. But the proposed DSPA algorithm can still guarantee the highest performance.

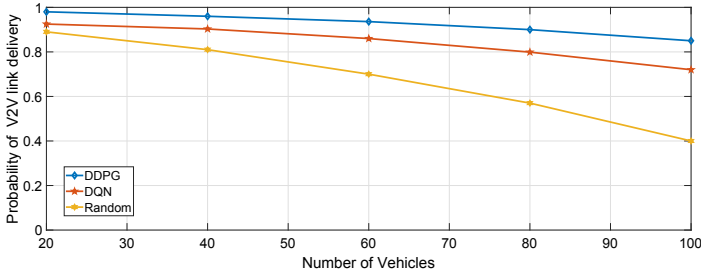


Fig. 6. Average delivery probability of V2V with different number of vehicles

6 Conclusion

This article studies the spectrum and power allocation issues in IoV. In order to ensure different QoS requirements in V2X communication, we model the spectrum reuse and power selection problem as a MDP. Since the action space is continuous, we design a DSPA method based on DDPG to solve the established MDP model. The simulation results show that the DPA algorithm can guarantee the high reliability of V2V communication and the high rate requirements of V2I communication. In future work, on the basis of ensuring QoS requirements, we will consider the issue of energy efficiency.

Acknowledgments. This work is partly supported by the National Natural Science Foundation of China (No. 61872044), Beijing Municipal Program for Excellent Teacher Promotion (No. *PXM2017_014224.000028*), The Key Research and Cultivation Projects at Beijing Information Science and Technology University (No. 5211823411).

References

1. Yun, D.S., Lee, S., Kim, D.H.: A study on the architecture of the in-vehicle wireless sensor network system. In: 2013 International Conference on Connected Vehicles and Expo (ICCVE), Las Vegas, NV, pp. 826–827 (2013). <https://doi.org/10.1109/ICCVE.2013.6799907>

2. Kombate, D., Wanglina: The internet of vehicles based on 5G Communications. In: 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (Smart-Data), Chengdu, pp. 445–448 (2016). <https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2016.105>
3. Peng, H., Liang, L., Shen, X., Li, G.Y.: Vehicular communications: a network layer perspective. *IEEE Trans. Veh. Technol.* **68**(2), 1064–1078 (2019). <https://doi.org/10.1109/TVT.2018.2833427>
4. Ioannou, I., Vassiliou, V., Christophorou, C., Pitsillides, A.: Distributed artificial intelligence solution for D2D communication in 5G networks. *IEEE Syst. J.* **14**(3), 4232–4241 (2020). <https://doi.org/10.1109/JSYST.2020.2979044>
5. Lai, W., Wang, Y., Lin, H., Li, J.: Efficient resource allocation and power control for LTE-A D2D communication with pure D2D model. *IEEE Trans. Veh. Technol.* **69**(3), 3202–3216 (2020). <https://doi.org/10.1109/TVT.2020.2964286>
6. Ibrahim, A., Ngatched, T.M.N., Dobre, O.A.: Optimal interference management, power control and routing in multihop D2D cellular systems. In: *IEEE Wireless Communication Networking Conference (WCNC)*, Marrakesh, Morocco, pp. 1–8 (2019). <https://doi.org/10.1109/WCNC.2019.8885752>
7. Liu, X., Chen, X., Chen, Y., Li, Z.: Deep learning based dynamic uplink power control for NOMA ultra-dense network system. In: Zheng, Z., Dai, H.-N., Tang, M., Chen, X. (eds.) *BlockSys 2019*. CCIS, vol. 1156, pp. 774–786. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2777-7_64
8. Gao, L., Hou, Y., Tao, X., Zhu, M.: Energy-efficient power control and resource allocation for V2V communication. In: *IEEE Wireless Communications and Networking Conference (WCNC)*, Seoul, Korea (South), pp. 1–6 (2020). <https://doi.org/10.1109/WCNC45663.2020.9120612>
9. Chen, Y., Yuan, H., Li, Y.: Object-oriented state abstraction in reinforcement learning for video games. In: *2019 IEEE Conference on Games (CoG)*, London, United Kingdom, pp. 1–4 (2019). <https://doi.org/10.1109/CIG.2019.8848099>
10. Luong, N.C., et al.: Applications of deep reinforcement learning in communications and networking: a survey. *IEEE Commun. Surv. Tutor.* **21**(4), 3133–317 (2019). Fourthquarter. <https://doi.org/10.1109/COMST.2019.2916583>
11. Tang, C., Chen, X., Chen, Y., Li, Z.: dynamic resource optimization based on flexible numerology and Markov decision process for heterogeneous services. In: *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, Tianjin, China, pp. 610–617 (2019). <https://doi.org/10.1109/ICPADS47876.2019.00092>
12. Liang, L., Ye, H., Li, G.Y.: Spectrum sharing in vehicular networks based on multi-agent reinforcement learning. *IEEE J. Sel. Areas Commun.* **37**(10), 2282–2292 (2019). <https://doi.org/10.1109/JSAC.2019.2933962>
13. Ma, T., Chen, X., Ma, Z., Chen, Y.: Deep reinforcement learning for pre-caching and task allocation in internet of vehicles. In: *2020 IEEE International Conference on Smart Internet of Things (SmartIoT)*, Beijing, China, pp. 79–85 (2020). <https://doi.org/10.1109/SmartIoT49966.2020.00021>
14. Silver, D., et al.: Deterministic policy gradient algorithms (2014)



A Pufferfish Privacy Mechanism for the Trajectory Clustering Task

Yuying Zeng^(✉), Yingpeng Sang^(iD), Shunchao Luo, and Mingyang Song

School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
sangyp@mail.sysu.edu.cn, {zengyy26, luoshch5, songmy5}@mail2.sysu.edu.cn

Abstract. Monitoring people's trajectories can help with many data mining tasks. For example, clustering analysis of users' trajectories can infer where people work and where they live. However, as far as users are concerned, no one wants their privacy to be compromised so that third parties can benefit from it. The commonly used method of privacy protection today is differential privacy, but differential privacy does not have significant advantages when dealing with correlated data. Pufferfish privacy can be used to address the privacy protection of correlated data for this reason. Our work aims to protect the locations that are extracted from trajectories using clustering methods. To achieve this goal, we first use the DBSCAN algorithm to cluster the trajectories of users, mark the locations where users have stayed, and preserve the chronological order. Privacy requirements for this issue are then specified in the Pufferfish framework. The data is then processed using a mechanism that implements a privacy framework. Finally, the utility is evaluated through experiments.

Keywords: Trajectory · DBSCAN · Correlated data · Pufferfish privacy · Markov chain

1 Introduction

In the age of big data, people's trajectories are becoming more accessible than ever before. These trajectories, when mined, can be useful in many ways, such as knowing where people live and where they work. However, it should not be ignored that this useful information is also sensitive for users, and they do not want their privacy to be disclosed. Therefore, how to obtain useful information under the premise of protecting users' privacy has become a hot topic. In the process of working with trajectories, it is a common research direction to obtain where users have stayed. Through these location sequences, we can know the users' daily routines and frequently visited locations. In this case, the user's trajectory from one location to another is of little use to data consumers and results in more privacy leaks. For this scenario, we can use a clustering algorithm to extract the locations where users have stayed from their trajectories and keep the time sequences of users passing through these locations. Then our protected

objects will become these sequences. Our method can protect the privacy of users' trajectories and facilitate the implementation of such data mining tasks.

Places where users have stayed are also the privacy of users, so when the locations are extracted from users' trajectories, the data still need to be processed to meet the privacy protection requirements. The concept of differential privacy [1] proposed by Dwork is still the gold standard of data privacy at the moment. However, differential privacy may not solve the privacy problem of the correlated data, as the association can lead to additional privacy leakage and lower the expected privacy guarantee [2]. Based on differential privacy, the concept of group differential privacy [3,4] is proposed for correlated data, but group differential privacy causes excessive interference with relevant data.

To solve the privacy protection problem of correlated data, we can adopt Pufferfish privacy [2] as a way to protect users' privacy. There are three parts for the definition of the Pufferfish framework: (S, Q, Θ) . S is for a set of secrets that cannot be known, such as "Alice is in the park"; Q refers to a set of secret pairs, such as ("Alice is in the park", "Alice is in the school"), and the adversary cannot distinguish between the secret pairs; Θ is the correlations of data, when data satisfy the distribution Θ , the secrets in Q are indistinguishable from each other to the adversary in the Pufferfish framework. The sequence of locations where a user has stayed follows the Markov chain model, and Pufferfish privacy can protect the privacy of correlated data [5], so Pufferfish privacy can be applied to this sequence to protect privacy.

The main contributions of our paper are as follows:

- We cluster trajectories before protecting the privacy of data, and that simplifies the privacy protection for trajectories.
- We consider the time correlation of data before publishing the locations where users have stayed, and use the Pufferfish privacy which has greater utility than group differential privacy in protecting the data for privacy. To the best of our knowledge, we are the first to apply the Pufferfish privacy in the trajectory privacy protection.

2 Related Work

Trajectory is a common form of private data, and because of the popularity of location-based services, it is widely used in data mining. Some useful information can be extracted from the trajectories [7], but trajectories contain users' privacy, so the process of extracting this valid information may cause users' privacy to be disclosed.

Clustering analysis is often used in the field of trajectory data mining [8]. According to the results of clustering analysis, the daily routines of users can be found, so that deeper information can be mined by the adversary. Since trajectories are private data, privacy protection should also be provided for data during data mining. In clustering analysis, the most common privacy protection technologies include data disturbance [9], data rotation [10] and data swapping [11].

There have also been previous researches on differential privacy clustering methods [12, 13]. In the k-means clustering algorithm, calculating the center point closest to each sample point will reveal privacy. To achieve privacy protection, noise satisfying differential privacy was added to the center point of each step. This kind of methods are not suitable for trajectory privacy protection for the following two reasons. Firstly, for trajectories mining, the k-means algorithm has worse performance than the density-based clustering. Secondly, when the clustering results are time-related, there will still be privacy disclosure issues [14]. Group differential privacy is an improvement over differential privacy. When data is related, sensitivity will be set to the size of the correlated data collection. However, using group differential privacy to protect the privacy of clustered data will introduce much noise and greatly decrease the utility of data. Pufferfish privacy has less impact on the utility of data than group differential privacy, and it already has some applications to protect the privacy of correlated data. For example, Pufferfish privacy can be adopted to protect the privacy of time-series data such as physical activity measurements [5, 6] and web browsing behavior [14]. To the best of our knowledge, we are the first to apply it in the trajectory privacy protection.

3 Preliminaries

This section mainly introduces some of the basic concepts presented in this paper, including the DBSCAN algorithm, the Pufferfish privacy, the Laplace mechanism, Markov chain, etc.

3.1 DBSCAN Algorithm

The DBSCAN algorithm [15] is a common and effective density-based clustering algorithm. There are two parameters in the DBSCAN algorithm: Eps is the user-specified radius coefficient, and $MinPts$ is the user-specified threshold.

The DBSCAN algorithm divides points into core points, border points, and noise points. A point is a core point if the number of points within or on a given neighborhood is greater than or equal to $MinPts$. The size of the neighborhood is determined by the distance function and Eps . The border point falls near the core point or the boundary, but it is not the core point. A noise point is a point that is neither a border point nor a core point. For example, in Fig. 1, A, B and C are respectively a core, border and noise point. The DBSCAN algorithm has a strong anti-noise capability and can handle clusters of arbitrary size and shape, and has many clustering results that cannot be obtained by k-means and other clustering algorithms.

3.2 Pufferfish Privacy

There are three parts for the definition of the Pufferfish framework: (S, Q, Θ) . S is for a set of secrets containing sensitive information that cannot be made

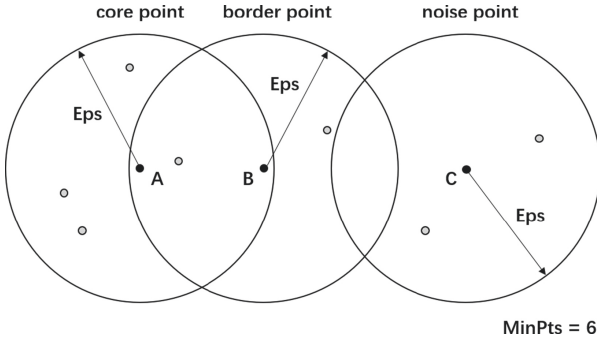


Fig. 1. Graphical representation of the core, border and noise points.

known to the adversary; Q refers to a set of secret pairs, and the adversary cannot distinguish between the secret pairs; Θ is the correlations of data which is in the hands of the adversary. When data satisfy the distribution Θ , the secrets in Q are indistinguishable from each other to the adversary in the Pufferfish framework. From the definition of Pufferfish privacy, we can see that differential privacy is a special case of Pufferfish privacy.

This privacy mechanism can be considered to satisfy the ϵ -Pufferfish privacy if the following conditions are met:

$$e^{-\epsilon} \leq \frac{P_{M,\Theta}(M(D) = w|s_i, \Theta)}{P_{M,\Theta}(M(D) = w|s_j, \Theta)} \leq e^\epsilon \tag{1}$$

M is a privacy mechanism that satisfies the ϵ -Pufferfish privacy with a range \mathcal{M} , $w \in \mathcal{M}$, ϵ is the privacy budget, D satisfies the distribution Θ . s_i and s_j is a pair of secrets, $P(s_i|\Theta) \neq 0$ and $P(s_j|\Theta) \neq 0$.

3.3 Laplace Mechanism

The Laplace distribution is a continuous probability distribution. When the probability density function of a random variable x is as follows, it is a Laplace distribution:

$$f(x|\mu, b) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}} \tag{2}$$

μ is the position parameter and $b > 0$ is the scale parameter.

Since differential privacy can be achieved by adding random noise satisfying Laplace distribution in the query $F(D)$, Pufferfish privacy can also be combined with the Laplace mechanism.

$$M(D) = F(D) + Z \sim Lap\left(\frac{\Delta f}{\epsilon}\right) \tag{3}$$

Δf refers to sensitivity, and the greater the Δf , the greater the noise.

3.4 Markov Chain

Markov chain is a discrete event stochastic process with Markov property in exponential science. Markov chain contains the first state and transition matrix. In each step, the system can change from one state to another or remain in the current state according to the transition matrix. Transitions represent state changes, and transition probabilities represent the probabilities associated with state changes. The Markov chain is not only one-dimensional but also multi-dimensional [16].

Suppose that the user has three frequented locations, A, B, and C. The user’s state transition diagram for these three locations is shown in Fig. 2. From the state transition diagram, the state transition matrix can be obtained:

$$\begin{bmatrix} 0 & 0.5 & 0.5 \\ 0.2 & 0 & 0.8 \\ 0.3 & 0.7 & 0 \end{bmatrix}.$$

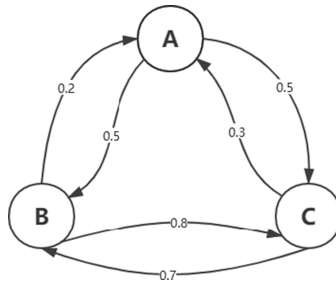


Fig. 2. The user’s state transition diagram.

4 A Pufferfish Privacy Mechanism for Trajectories Processed by the DBSCAN Algorithm

Our Pufferfish privacy mechanism for trajectory clustering consists of two steps. The first step is to use the DBSCAN algorithm to obtain the sequence of locations where the user has stayed, and the second step is to use the Laplace mechanism to process the sequence so that the sequence satisfies Pufferfish privacy.

Algorithm 1. Get the sequence of locations and preserve they in chronological order.

Input: T : trajectory data set; Eps : the user-specified radius coefficient; $MinPts$: the user-specified threshold; v : the user-specified value.

Output: P : a sequence of locations that preserve the chronological order.

```

1: while there are unlabeled points do
2:   Check whether the unlabeled point  $p$  in  $T$  is a core point;
3:   if  $p$  is not a core point then
4:     Label  $p$  as a noise point;
5:   else
6:     Label  $p$  as a core point;
7:     A new cluster  $C_{new}$  is formed using  $p$  and includes all the points within the
       $Eps$  neighborhood or on the boundary within the cluster;
8:     Insert these points into the queue;
9:     while the queue is not empty do
10:      Delete the first point  $p_{first}$  in the queue;
11:      if  $p_{first}$  is not a core point then
12:        Label  $p_{first}$  as a border point;
13:      else
14:        Label  $p_{first}$  as a core point;
15:        Store all neighbor points that not assigned a category of  $p_{first}$  as
         $P_{neighbor}$ ;
16:        for  $p_{neighbor}$  in  $P_{neighbor}$  do
17:          Assign  $p_{neighbor}$  to the current category  $C_{new}$ ;
18:          Insert  $p_{neighbor}$  into the queue;
19:        end for
20:      end if
21:    end while
22:  end if
23: end while
24: for each trajectory  $T_i$  in  $T$  do
25:   Get the category sequence  $C_i$  in chronological order;
26:   for each category label  $c$  in  $C_i$  do
27:     Count the number of consecutive occurrences of  $c$ , save it as  $cnt$ ;
28:     if  $cnt < v$  then
29:       Delete the consecutive occurrences of  $c$  from  $C_i$ ;
30:     end if
31:   end for
32: end for
33: Clear adjacent and identical data in  $P$ ;
34: return  $P$ .

```

4.1 Acquisition of the Sequence of Frequently Visited Locations

A trajectory is composed by a sequence of data points in the 3-dimensional space including the latitude, longitude and time dimension. We firstly ignore the time information and use the DBSCAN algorithm to cluster the latitude and longitude information. After the clustering result is obtained, we arrange

the locations where the user has stayed according to the time sequence and retain the locations where the user has stayed repeatedly.

We describe the entire process in Algorithm 1. Step 1 to Step 23 is to cluster trajectory data set using the DBSCAN algorithm, and after clustering, every point on the trajectories except the noise points is classified into a category. Step 24 to Step 32 is to get the sequence of frequently visited locations. We observe the number of consecutive occurrences of each category. If a category appears less than v , a parameter for measuring location visiting frequency, it will be deleted. After deletion, if the categories before and after the deleted one are the same, they will be merged.

4.2 Pufferfish Privacy for Location Privacy Protection

The mechanism of Pufferfish privacy is closely related to the query F . We use a mechanism that can be applied to any general Pufferfish instantiation. The mechanism defines the Laplace mechanism for Pufferfish privacy and uses Wasserstein Distance as the sensitivity to process the data. The query F of the mechanism maps X into a scalar.

Definition 1 (Wasserstein Distance). Wasserstein distance measures the distance between two probability distributions (see Fig. 3). It can be defined as follows:

$$W(P_\mu, P_\nu) = \inf_{\gamma \sim \Pi(P_\mu, P_\nu)} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|] \tag{4}$$

$\Pi(P_\mu, P_\nu)$ is the set of all possible joint distributions of P_μ and P_ν . Firstly, we can sample $(x, y) \sim \gamma$ to get a true sample x and a generated sample y for each possible joint distribution γ . Secondly, we calculate the distance $\|x - y\|$ from the pair of samples. At last, we can calculate the mean $\mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|]$ of the sample pairs under this joint distribution γ . The lower bound that can be taken from this mean of all possible joint distributions is defined as Wasserstein distance.

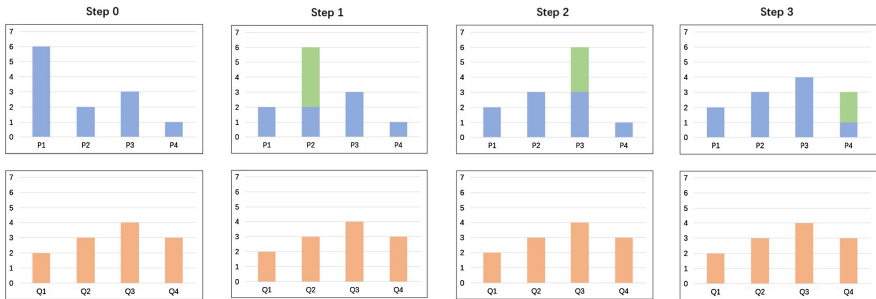


Fig. 3. The process from distribution P to distribution Q .

So, this mechanism can be expressed as:

$$M(D) = F(D) + Z \sim Lap\left(\frac{W(P_\mu, P_\nu)}{\epsilon}\right) \tag{5}$$

We can describe our mechanism in Algorithm 2.

Algorithm 2. A Pufferfish privacy mechanism for location privacy protection.

Input: X : a sequence of sites with correlations; (S, Q, Θ) : Pufferfish framework; F : query function; ϵ : privacy budget.

Output: $F(X) + Z$

- 1: **for** all $(s_i, s_j) \in S$ and $\theta \in \Theta(P(s_i|\theta) \neq 0$ and $P(s_j|\theta) \neq 0)$ **do**
 - 2: Set $\mu = P(F(X) = w|s_i, \theta), w \in \mathcal{M}$;
 - 3: Set $\nu = P(F(X) = w|s_j, \theta), w \in \mathcal{M}$;
 - 4: Calculate $W(\mu, \nu)$;
 - 5: **end for**
 - 6: Set $W = \sup_{(s_i, s_j) \in Q, \theta \in \Theta} W(\mu, \nu)$;
 - 7: **return** $F(X) + Z \sim Lap\left(\frac{W}{\epsilon}\right)$.
-

Example. Consider a Pufferfish instantiation of location privacy protection. Suppose that the user has three frequently visited locations, A, B, and C. The user’s current location is L_i , and the user’s next location is L . We have:

L_i	$P(L = A)$	$P(L = B)$	$P(L = C)$
A	0	0.5	0.5
B	0.2	0	0.8
C	0.3	0.7	0

Suppose in this data set, the location of the user at the present moment is a secret. The current location and the location of the other form a secret pair. Let’s assume that A is the current location, and its secret pair is B. We will predict that the next location is L_j and then we can get the probability distribution for the next location:

L_j	A	B	C
$P(L = L_j A)$	0	0.5	0.5
$P(L = L_j B)$	0.2	0	0.8

The Wasserstein distance between these two distributions is 0.1. If we use group differential privacy, the sensitivity mechanism would add $Lap(3/\epsilon)$ noise, which gives a worse utility.

5 Experiments

The dataset we used is the Geolife GPS Trajectories dataset [17]. In this paper, we used the longitude, latitude, and time information from that dataset to determine the user’s location. The parameters Eps , $MinPts$ and v are respectively 0.0003, 80 and 10 in Algorithm 1 to get the sequence of locations in the chronological order. After completing the clustering task, we get a sequence of locations for each user in each day of a period of time, which are our privacy protection objects. The query function returns the next location given a current location, and the number of queries is 1000. We compare with group differential privacy to demonstrate the usefulness of our mechanism based on the Pufferfish framework, by evaluating the utilities of these two methods. We use the L_1 loss as a measure of utility, which is defined as follows:

$$L_1\text{-Loss} = \frac{\sum_{i=1}^n |f(x_i) - y_i|}{n}, \tag{6}$$

in which $f(x_i)$ and y_i are respectively the query result and true result of the i -th query, and n is the number of queries. Finally, we obtain the results as shown in Fig. 4.

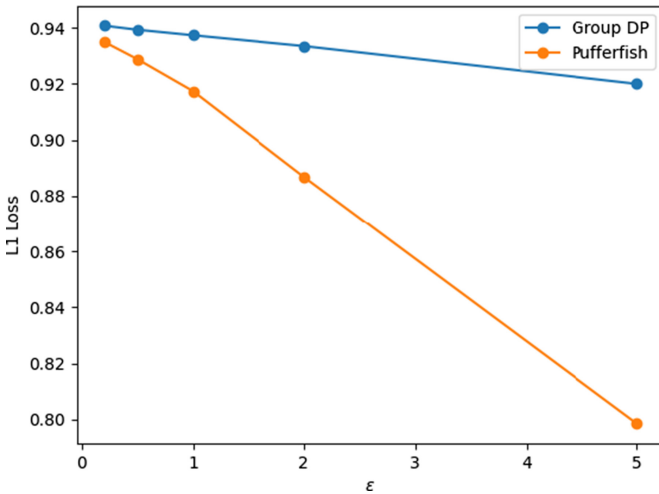


Fig. 4. $L_1\text{-Loss}$ vs ϵ .

As we can see from the results, our mechanism achieves greater utility than group differential privacy when the privacy budget is the same.

6 Conclusion

We combine the DBSCAN algorithm with Pufferfish privacy to provide a Pufferfish privacy mechanism for trajectory privacy protection. Typically, we can

cluster trajectories to get the locations where users have been frequently visited. However, if we directly publish these related sequences (i.e., locations in their chronological order), they will very easily cause privacy leakage. So, we propose our mechanism to protect users' privacy in these sequences. Experiments show that our mechanism achieves greater utility than group differential privacy.

In the future, we can improve our work in the following ways. We can mine correlations in trajectory data utilizing models other than Markov chain. At the same time, different privacy mechanisms can be developed by defining different query functions.

Acknowledgement. This work was supported by the Key-Area Research and Development Program of Guangdong Province (No. 2020B010164003), the Science and Technology Program of Guangzhou, China (No. 201904010209), and the Science and Technology Program of Guangdong Province, China (No. 2017A010101039). The corresponding author is Yingpeng Sang.

References

1. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006). https://doi.org/10.1007/11681878_14
2. Kifer, D., Machanavajjhala, A.: Pufferfish: a framework for mathematical privacy definitions. *ACM Trans. Database Syst.* **39**(1), January 2014. <https://doi.org/10.1145/2514689>
3. Dwork, C., Roth, A.: The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9(3–4), 211–407 (2014). <https://doi.org/10.1561/04000000042>
4. Fan, L., Xiong, L.: An adaptive approach to real-time aggregate monitoring with differential privacy. *IEEE Trans. Knowl. Data Eng.* **26**(9), 2094–2106 (2014)
5. Song, S., Wang, Y., Chaudhuri, K.: Pufferfish privacy mechanisms for correlated data. In: *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD 2017*, pp. 1291–1306. Association for Computing Machinery, New York, NY, USA (2017). <https://doi.org/10.1145/3035918.3064025>
6. Xi, Z., Sang, Y., Zhong, H., Zhang, Y.: Pufferfish privacy mechanism based on multi-dimensional Markov chain model for correlated categorical data sequences. In: *Parallel Architectures, Algorithms and Programming - 10th International Symposium, PAAP 2019, Guangzhou, China, 12–14 December, 2019, Revised Selected Papers*, pp. 430–439 (2019). https://doi.org/10.1007/978-981-15-2767-8_38
7. Zheng, Y.: Trajectory data mining: an overview. *ACM Trans. Intell. Syst. Technol.* **6**(3), May 2015. <https://doi.org/10.1145/2743025>
8. Xiaowei Xu, Ester, M., Kriegel, H., Sander, J.: A distribution-based clustering algorithm for mining in large spatial databases. In: *Proceedings 14th International Conference on Data Engineering*, pp. 324–331 (1998)
9. Lindell, Y., Pinkas, B.: Privacy preserving data mining. In: Bellare, M. (ed.) *CRYPTO 2000*. LNCS, vol. 1880, pp. 36–54. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-44598-6_3
10. Oliveira, S.R.M., Zaïane, O.R.: Achieving privacy preservation when sharing data for clustering. In: Jonker, W., Petković, M. (eds.) *SDM 2004*. LNCS, vol. 3178, pp. 67–82. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30073-1_6

11. Fienberg, S.E., McIntyre, J.: Data swapping: variations on a theme by dalenius and reiss. In: Domingo-Ferrer, J., Torra, V. (eds.) PSD 2004. LNCS, vol. 3050, pp. 14–29. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-25955-8_2
12. Blum, A., Dwork, C., McSherry, F., Nissim, K.: Practical privacy: the sulq framework. In: Proceedings of the Twenty-Fourth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2005, pp. 128–138. Association for Computing Machinery, New York (2005). <https://doi.org/10.1145/1065167.1065184>
13. Dwork, C.: A firm foundation for private data analysis. Commun. AC, January 2011. <https://www.microsoft.com/en-us/research/publication/a-firm-foundation-for-private-data-analysis/>
14. Liang, W., Chen, H., Liu, R., Wu, Y., Li, C.: A pufferfish privacy mechanism for monitoring web browsing behavior under temporal correlations. Comput. Secur. **92**, 101754 (2020). <http://www.sciencedirect.com/science/article/pii/S0167404820300389>
15. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise, pp. 226–231. AAAI Press (1996)
16. Zhang, W., Zhao, J., Wei, F., Chen, Y.: Differentially private high-dimensional data publication via markov network. EAI Endorsed Trans. Secur. Saf. **6**(19), January 2019
17. Asia, M.R.: Geolife gps trajectories. <https://www.microsoft.com/en-us/download/details.aspx?id=52367> Last published: August 9, 2012



A Novel Attention Model of Deep Learning in Image Classification

Qiang Hua, Liyou Chen, Pan Li^(✉), Shipeng Zhao, and Yan Li^(✉)

Key Laboratory of Machine Learning and Computational Intelligence of Hebei Province, College of Mathematics and Information Science, Hebei University, Baoding 071002, China
1173688949@qq.com, ly@hbu.edu.cn

Abstract. As the neural network becomes more and more complex, a large number of parameters are to be adjusted and more unrelated information will be generated, which is more time-consuming in model training and affects the model performance. The attention mechanism is often used to address this problem in the literature. However, the current attention models only consider the correlation within the pixel domain and channel domain individually, and the computational complexity is comparatively high. This paper proposes a new type of attention module called Pixel-wise And Channel-wise attention (PAC attention module for short) in which the hybrid correlation between the pixel and channel domain is also considered. Without increasing the number of parameters, this module can be used to calculate the correlation of the convolution feature map at any position in any layer of the convolutional neural network to achieve end-to-end training. In the experiments, the proposed PAC attention module is used in typical network structures in deep learning and has been compared to current models on some benchmark big datasets.

Keywords: Attention mechanism · Deep learning · Image classification · Convolutional neural networks

1 Introduction

Motivated by human attention mechanism theories, the attention mechanism [1, 2] is a type of models that learns to focus on important information while ignoring irrelevant information. Based on this idea, different attention models (AM) are proposed to reduce the complexity of learning models and at the same time enhance the performance. In the literature, AM has been successfully applied in many fields such as image processing, document classification, voice recognition, computer vision, machine translation, etc. [3–9]. In deep neural network, the attention mechanism is expressed as a parameter allocation scheme, assigning larger parameters to the areas we need to focus on can help alleviate the loss of model information, such as long-range dependence. In 2014, Google Deep-Mind team proposed a multiple-stage hard attention mechanism for Recurrent Neural Network (RNN) and the method could decrease the error rate obviously in image classification task on MINIST dataset [10, 11]. In recent years, AM has become one of the research focus in deep learning, which can balance the

complexity and the representation capacity of the network models. For example, in the structure design of Convolutional Neural Network (CNN), the attention mechanism has been considered to introduce the receptive field and local connection, thus reducing the number of parameters while enhancing the representation capability [12–17]. Zhang et al. developed a part based region CNN based on the hard attention mechanism for the task of object detection [18]. A two level attention method was proposed by Xiao et al. [19] which used the hard attention of both object level and the part level without using any extra part information. Since the output function of the hard attention is not derivative, soft attention mechanism is then introduced for computational convenience. Fu et al. proposed the Recurrent attention CNN(RA-CNN) using both the hard and soft attention and then the attention weight is derivative and the end-to-end training network model can be generated [20]. Most of the above methods only used the pixel attention and ignored the attention on channel dimension. In 2017, Hu et al. developed the Squeeze-and-excitation network (SENet) for CNN which focuses on the channel relationship and adaptively recalibrates channel-wise feature responses by explicitly modelling interdependencies between channels [13]. Besides CNN, there are also attention models developed for other network architecture. In 2018, Wang et al. [14] introduced a Non-local module for capturing long-range image dependencies and can be applied to other computer vision architectures. SAGAN [21] is a combination of the Non-Local module example, which can be used to generate high-quality images.

To summarize, in the application of image classification, the attention learning models have been proposed based on individual domain, e.g., spatial domain [12], channel domain [13], and pixel domain[14], etc. There is less research work related to attention model in hybrid domain [15, 16] due to the obvious increase of the number of parameters causing much more storage and computation load. Therefore, it is a significant and challenging work to develop attention models in hybrid domain which can grasp the most important information while not increasing the model complexity noticeably. In this paper, a novel attention model called PAC model is developed, and the computation method in each module is analyzed in detail. The correction among pixels can be learned in CNN based on the module, and meanwhile the correlation between channels can be also learned. Furthermore, the PAC attention module can be combined with CNN and implemented end to end. Extensive experiments have been conducted to apply the proposed model in image classification problems, and the results have been analyzed in detail.

2 Preliminary

In this section, two related attention models, SENet [13] and Non-local Network [14] are briefly introduced. According to the type of attended region, the attention models can be divided into the models on spatial domain, channel domain and mixed domain [15], etc. Another typical attention mechanism is called self-attention mechanism which can be implemented between pixels as well as channels, learning the correlation among the feature mappings. The most often used attention mechanism between channel regions is SENet, while between pixels is Non-local Network.

2.1 Attention Model on Channel Domain: SENet

SENet learns the correlation between channels in image features, and attempt to select the most important feature channels related to the current task. Specifically, its computation process is as follows.

Suppose the input feature vector X is in size $H' \times W' \times C'$. Firstly, the output feature vector U in size $H \times W \times C$ can be obtained through convolution computation of U . Then the filter response is adjusted through squeezing and excitation processes. The squeezing process is done by taking the global average pooling of U :

$$z_c = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W u_c(i, j) \quad (1)$$

where z_c is the outputs after squeezing an individual channel; u_c is a certain channel dimension of U and $u_c \in R^{H \times W}$. The results after squeezing is z with the size of $1 \times 1 \times C$, which ignores the spatial information to help discover the relationship of different channels.

Based on the squeezing output, the excitation process is to learn the correlation among channels, which can be realized through a 2-layer network with full connections:

$$s = \sigma(W_2 \delta(W_1 z)) \quad (2)$$

where $W_1 \in R^{C/r \times C}$, $W_2 \in R^{C \times C/r}$ denote the weights of the two full connected layers; r is a scaled parameter to reduce the computation load. $\sigma(\cdot)$ is the Sigmoid function and $\delta(\cdot)$ the ReLU function.

Finally, the learned correction among channels can be considered as the weights which can be assigned to the original input features.

2.2 Attention Module in Pixel Domain: Non-local Network

Wang et al. [14] proposed the Non-local Neural Network based on the self-attention mechanism in natural language processing [22]. It is an effective, simple and general-purpose component for capturing the long-range dependence between pixels in image features in deep neural networks, and is often used as a self-attention model in the field of image processing [23].

Suppose that $X = [x_1, x_2, \dots, x_N]$ denotes N inputs. To reduce the computation cost, not all the N input vectors are necessary to be inputted in the neural network. Instead, those input vectors related to the task are selected to be inputted to the neural network. Generally, the self-attention mechanism can be described the following formula:

$$y_i = \frac{1}{C(x)} \sum_{\forall j} f(x_i, x_j) g(x_j) \quad (3)$$

where $i, j \in \{1, 2, \dots, N\}$, i is the input index and j take the values of indexes of all possible input locations. Since i and j generally represent the pixel location of image data, this attention model is called as attention model in pixel domain. x is the input signal which is often the features of images, sequences or videos etc.; y is the output signal with the same size of x . The pairwise function computes all the possible pair of locations i and j which is a scalar representing the relationship of different locations. The one variable function g calculates the representation of input signal at position j . The final output y is normalized using $C(x)$ by weighted average.

This attention mechanism can be used to construct the relationship between non adjacent positions, and the convolution layer itself can be used to construct the relationship of local information. Therefore, embedding this attention mechanism in convolution neural network can combine the non-local and local information, which enables us to build a richer hierarchical structure. Furthermore, the attention operation is a flexible building block, which can be added to the early part of convolutional neural network.

3 A Novel Attention Model in Image Classification

In convolutional neural network for image processing, the common attention module only measures the correlation of pixel features or feature channels individually, but ignores the relationship between them (pixel features and feature channels), resulting in the possible loss of attention. Thus, the attention obtained from the features extracted by the convolutional neural network cannot accurately express the attention of the original image. To solve this problem, we propose a Pixel-wise And Channel-wise Attention (PAC attention) mechanism. As a module, this mechanism can be embedded into the convolutional neural network model, thus increasing the representation ability of the model. This section first introduces the network structure of PAC attention module, and then briefly analyzes different attention modules, and verifies the performance of PAC attention module in image classification model.

3.1 The Structure of PAC Module

Suppose the feature size of the input image is $x \in R^{C \times H \times W}$, where C representing the number of channels of the feature; W representing the width of the feature; and H representing the height of the feature. According to different channels, the feature map x can be divided into $G = \{g_1, g_2, \dots, g_i, \dots, g_c\}$, where g_i refers to the i -th feature map in the feature map.

The dual correlation module considering both pixel and channel consists of three parts: the first is the pixel attention used to measure the correlation between pixels; the second part is channel attention used to calculate the correlation between channels; and the third is the mixed module to integrate the results of the above two by matrix multiplication. The features obtained by this mixed operation include not only the correlation in the feature dimension but also in the pixel dimension. The overall structure of PAC model is shown in Fig. 1.

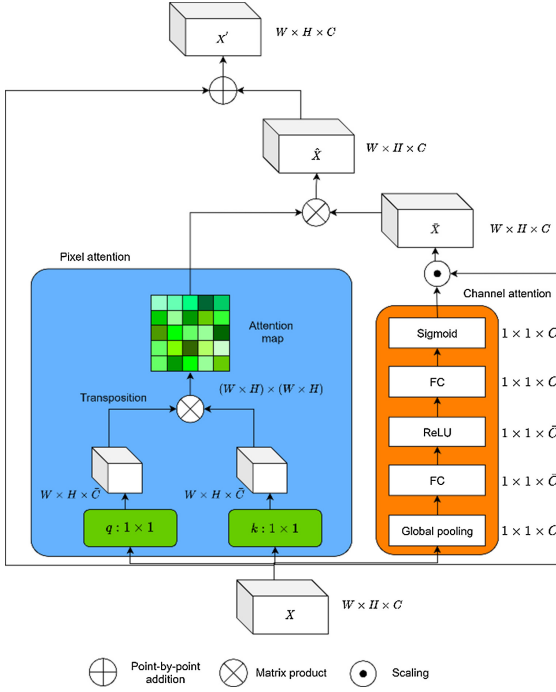


Fig. 1. The structure of PAC model

3.2 Pixel Attention

In CNN, because the size of each convolution kernel is limited, each convolution operation can only cover a small neighborhood around the pixel. Moreover, the convolution operation only adds the weighted local regions of input data. In order to capture the long-distance dependence of images and their features, it is necessary to calculate the affinity between pixels x_i and x_j . Before calculating the affinity between pixels, it is necessary to transform the image features from the previous hidden layer.

Firstly, using two feature spaces Q and K to linearly transform the inputted image feature $x \in R^{C \times H \times W}$ to reduce the computational complexity of the model:

$$q(x) = W_q x, k(x) = W_k x \tag{4}$$

where $W_q \in \bar{C} \times C, W_k \in \bar{C} \times C$ is the weight matrix to be learned through $1 \times 1 \times 1$ convolution operation. Secondly, the correlation between x_i and x_j can be obtained by the product computation:

$$r_{ij} = q(x_i)^T k(x_j) \tag{5}$$

Finally, the normalized result can be computed by the Softmax function:

$$\alpha_{ij} = \frac{\exp(r_{ij})}{\sum_{i=1}^N \exp(r_{ij})}$$

where α_{ij} represents the correlation between pixels x_i and x_j . C is the number of channels; \overline{C} reduces the number of C to $1/k$ of its original which finally reduces the complexity of correlation computation among pixels. W represents the width of the image features; H the depth of the image features; $N = H \times W$ is the number of pixels in an individual feature.

To summarize the correlation among all pixels, the pixel correlation matrix can be obtained as $A = (\alpha_{ij})_{N \times N}$, which is also called attention mapping or attention map.

3.3 Channel Attention

In CNN, hidden layers are usually invisible. However, through the visualization of hidden features, it can be seen that the same hidden layer can learn some combination of image features, some of the initial hidden layers can learn some meaningful edges, and deeper hidden layers can learn more representative contours, etc. However, a single hidden unit cannot construct the description of the whole image, which requires the overall analysis of different hidden layer units of the same hidden layer, that is, the correlation analysis between feature channels. By obtaining the correlation between channels, the sensitivity of network model to task related information can be improved.

In order to obtain the correlation of image channels, it is necessary to compress the signals of each input channel to obtain the whole reference symbol of the channel denoted by $s, s \in R^C$. s is generally represented by the channel statistical data generated by global average pooling, where s_i is the channel feature map of the i -th channel, i.e., the statistical data of g_i , which is computed as:

$$s_i = \frac{1}{N} \sum_{j=1}^N x_{i,j} \quad (7)$$

where $x_{i,j}$ denotes the data of the j -th pixel in the i -th channel feature map. The global average pooling method is convenient and efficient. Of course, we can also use the second-order pooling method to obtain channel correlation [24].

In order to obtain the overall information of each channel by global average pooling, it is necessary to re-calibrate it to learn the correlation between channels. For the sake of simplicity, we can use multi-layer perceptron (e.g., a three-layer network) to learn autonomously through the network to achieve the attention weight of channel domain:

$$w = \sigma(W_2 \delta(W_1 s)) \quad (8)$$

where δ denotes ReLU function, σ is the Sigmoid function, $W_1 \in R^{\bar{C} \times C}$, $W_2 \in R^{C \times \bar{C}}$, \bar{C} is used to reduce the computational complexity of the model.

Finally, the obtained attention weights of channel domain need to be applied to the original input feature map to adjust the importance of different channel features. In fact, the learned activation value of each channel is applied to the original input feature in the form of dot multiplication:

$$\bar{x} = w_c \cdot g_c \quad (9)$$

3.4 Mixed Module

Through the pixel dimension attention module, the correlation between pixels in the original feature map can be obtained, and the channel dimension attention module is used to add the correlation between channels to the original feature map. After that, in the mixed module, the matrix multiplication is used to aggregate them, which is equivalent to adding both the influence of channel correlation and pixel correlation to the original feature map.

Firstly, the dimension of $\bar{x} \in R^{C \times H \times W}$ is transformed to $\bar{x} \in R^{C \times N}$, where $N = H \times W$. The purpose of this transformation is to obtain a new feature map by matrix multiplication with pixel correlation matrix A :

$$\hat{x} = \bar{x}A \quad (10)$$

Secondly, the computed $\hat{x} \in R^{C \times N}$ is re-transformed to the original dimension size $\hat{x} \in R^{C \times H \times W}$, and then this obtained \hat{x} contains the correlation of pixels as well as that of channels.

In addition, the obtained dual correlation feature map is multiplied by the scale parameter, and then it is spliced by the identity connection of the residual network as the output of the whole module. Therefore, the final output of PAC attention module is given by the following formula:

$$o = \gamma \hat{x} + x \quad (11)$$

where γ is the learning rate, which is initialized to zero. The reason for this initialization is that the model is expected to learn simple tasks first, and then gradually increase the complexity of tasks.

The feature size outputted by PAC attention module is consistent with the input feature size, which is convenient for end-to-end training and application in various network models.

3.5 The Analysis and Comparisons of Related Models

The idea of the proposed PAC attention model is similar to that of Non-local network [14] (see Sect. 2.2). However, Non-local network is only used to measure the long-range dependence between pixels, while PAC attention model also takes into account

the correlation of channel dimensions in image features, and does not increase the complexity of the model. Furthermore, the parameters of PAC attention model are significantly reduced. Non-local network uses $1 \times 1 \times 1$ convolution for feature mapping, and the MACCs (multiply-accumulate operations) of model is $W \times H \times C$, while the number of parameters of PAC model is $2 \times C \times \bar{C}$. Obviously, we have $2 \times \bar{C} < H \times W$.

The overall network architecture of DANet model [17] is the most similar to the model proposed in this paper. However, since its overall parameter quantity is the same as that of Non-local network, DANet is more complex than the proposed PAC model.

CBAM [16] model can be regarded as an extension of SENet [13]. Although the model also pays attention to spatial domain and channel domain, its calculation method in spatial domain is different from PAC attention model. In spatial domain, PAC attention model adopts an approximate second-order calculation method calculating the covariance matrix as the image representation, which can enhance the nonlinear modeling ability of the network [24]. In the channel domain, CBAM adopts the maximum pooling and average pooling, while PAC only adopts the global average pooling, which reduces the computation to great extent.

The comparisons of different related attention models are summarized in Table 1 in terms of parameter quantity, calculation amount and expression ability. The proposed PAC attention model has the characteristics of less parameters and lower computational complexity. In addition, PAC attention model maintains the modularity, and can be placed behind any convolution layer to reconstruct image features according to the relevance and importance of data points.

Table 1. The qualitative analysis of difference attention models

Model	SENet	CBAM	Non-Local	DANet	PAC
Parameters numbers	Minimum	Less	Maximum	Maximum	More
Computation load	Least	Less	More	Maximum	Middle
Representative ability	Weak	Relative weak	Middle	Relative strong	Strong

4 Experimental Results in Image Classification

4.1 Experimental Settings and the Used Datasets

We use CNN in the classification task and its basic structures and settings are as follows in Table 2, where *conv*() represents the convolution layer used to extract image features; *s* represents the convolution step size; *P* represents whether zero padding or not; and *out* represents the number of output characteristic channels. The normalization layer will use batch normalization (BN) [25] to standardize the data; pooling layer is used to reduce the data dimension, including max pooling, average pooling and global average pooling; dropout is a common operation to prevent over fitting in neural networks.

Table 2. The network structure of CNN in image classification

Input	Network	Output
$3 \times 32 \times 32$	conv(3×3 , $s = 1$, $p = 1$, out = 64)	$64 \times 32 \times 32$
$64 \times 32 \times 32$	conv(3×3 , $s = 1$, $p = 1$, out = 64)	$64 \times 32 \times 32$
$64 \times 32 \times 32$	max pooling(2×2 , $s = 2$)	$64 \times 16 \times 16$
Dropout		
$64 \times 16 \times 16$	conv(3×3 , $s = 1$, $p = 1$, out = 128)	$128 \times 16 \times 16$
$128 \times 16 \times 16$	conv(3×3 , $s = 1$, $p = 1$, out = 128)	$128 \times 16 \times 16$
$128 \times 16 \times 16$	avg pooling(2×2 , $s = 2$)	$128 \times 8 \times 8$
Dropout		
$128 \times 8 \times 8$	conv(3×3 , $s = 1$, $p = 1$, out = 256)	$128 \times 8 \times 8$
$128 \times 8 \times 8$	conv(3×3 , $s = 1$, $p = 1$, out = 256)	$128 \times 8 \times 8$
$128 \times 8 \times 8$	Global average pooling	$256 \times 1 \times 1$
256	Full-connection layer	10

Datasets: The benchmark image dataset Cifar-10 [26] is used in different attention models to test the classification performance, which is composed of 60000 color images with the size of 32×32 from 10 categories and each category has 6000 images. Another bigger image dataset CIFAR-100 [26] is used in the experiments when comparing the proposed PAC model with the current popular classification models. Cifar100 is similar to cifar10 dataset, but it contains more categories, including 100 types of image data, and the image categories of the cifar100 dataset are more refined.

4.2 Classification Performance Using Different Attention Models

According to the convolution neural network structure listed in Table 2, the experiment in this section is to apply our attention module and other attention modules to the first layer convolution and the second layer convolution. The experimental attention modules include PAC attention model and SENet model [13], Non-local Network [14], CBAM model [16], and DANet model [17].

We use different performance indexes to evaluate these models including the required number of parameters, FLOPs(floating point of operations), and the classification accuracy. FLOPs represent the total amount of computation needed to run the model, and are often used to measure the computational complexity of algorithms or models. The results of classification experiments are shown in Table 3, which show that compared with other attention modules, PAC attention module can achieve the highest classification accuracy.

Table 3. The classification results of different attention models using CNN

Models	Storage of parameters (Bit)	FLOPs	5 epoch Accuracy (%)	8 epoch Accuracy (%)	10 epoch Accuracy (%)
Original CNN	1148874	153466112	83.26	84.16	83.36
SENet	1153066	153613696	82.32	84.36	84.26
Non-local Network	1154074	164115712	82.82	84.39	84.99
CBAM	1151020	153806208	83.14	84.06	84.36
DANet	1154074	164115712	80.95	82.66	83.66
PAC	1154106	155743616	84.23	84.88	85.10

4.3 Performance Analyses of PAC Attention Module with Different Typical Deep Learning Models

In this section, we combine the proposed PAC module with VGG-16 and GoogleNet [27, 28] on dataset Cifar100 for the task of image classification, and compare the results with different popular classification deep learning models [13, 15, 27–32]. Table 4 shows all the results in terms of the number of parameters and the error rate, in which the number behind each model denote the number of layers in the neural network, e.g., VGG-16 has 16 layers. Compared with the experimental results of VGG -16 and VGG-16 + PAC, GoogleNet and GoogleNet + PAC, it can be seen that adding PAC attention module can improve the classification accuracy of the model without significant increase of parameters. Compared with other classification models, GoogleNet + PAC achieves the lowest Top-1 and Top-5 classification error rate, which is 21.7, 5.54%, respectively. This is a noticeable improve in classification performance without obviously increasing the number of parameters, which verifies the effectiveness of the proposed PAC model.

Table 4. The classification performance of different models on Cifar100

Models	Storage of parameters (M)	Top-1 error rate (%)	Top-5 error rate (%)
VGG-16 [2]	34.01	27.17	8.83
GoogleNet [3]	6.26	21.97	5.98
ResNet-34 [4]	21.32	23.34	6.73
SE-ResNet-34 [6]	21.65	22.07	6.12
Res-Attention59 [8]	55.75	33.62	12.98
DenseNet121 [27]	7.04	22.89	6.54
MobileNet [29]	3.34	34.12	10.57
Inception-V3 [31]	22.37	22.82	6.41
VGG-16 + PAC	34.04	26.97	8.72
GoogleNet + PAC	6.31	21.70	5.54

5 Conclusions

In this paper, PAC attention model is proposed, which can simultaneously pay attention to both the pixel domain and channel domain of image features, that is, it can detect the importance of each channel of image features while exploring the long-range dependence between image feature pixels. Extensive experiments are conducted according to the attention mechanism, and different attention models under the same convolution neural network architecture are carried out. Compared with other attention models, PAC module applied to image classification achieves higher classification accuracy on cifar10 dataset. It is found that PAC attention module can generate higher attention to the objects to be recognized in the classification task.

Acknowledgments. This research is supported by the Natural Science Foundation of China (61976141), the Natural Science Foundation of Hebei Province (F2018201096), the Natural Science Foundation of Guangdong Province (2018A0303130026) and the Key Foundation of the Education Department of Hebei Province (ZD2019021).

References

1. Treisman, A.M., Gelade, G.: A feature-integration theory of attention. *Cogn. Psychol.* **12**(1), 97–136 (1980)
2. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(11), 1254–1259 (1998)
3. Xu, K., Ba, J., Kiros, R., et al.: Show, attend and tell: Neural image caption generation with visual attention. In: *International Conference on Machine Learning. ICML 2015*, pp. 2048–2057 (2015)
4. Yang, Z., Yang, D., Dyer, C., et al.: Hierarchical attention networks for document classification. In: *Proceedings of the 2016 Conference of the North American CHAPTER of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489 (2016)
5. Xiao, L., Chen, B.L., Huang, X., et al.: Multi-label text classification method based on label se-mantic information. *J. Softw.* **31**(4), 1079–1089 (2020)
6. Chan, W., Jaitly, N., Le, Q., et al.: Listen, attend and spell: a neural network for large vocabulary conversational speech recognition. In: *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4960–4964. IEEE (2016)
7. Lu, J., Yang, J., Batra, D., et al.: Hierarchical question-image co-attention for visual question answering. In: *Advances in Neural Information Processing Systems*, pp. 289–297 (2016)
8. Yin, W., Schütze, H., Xiang, B., et al.: ABCNN: attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint [arXiv:1512.05193](https://arxiv.org/abs/1512.05193)* (2015)
9. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. *arXiv preprint [arXiv:1706.03762](https://arxiv.org/abs/1706.03762)* (2017)
10. Mnih, V., Heess, N., et al.: Recurrent models of visual attention. *arXiv preprint [arXiv:1406.6247](https://arxiv.org/abs/1406.6247)* (2014)
11. Ba, J., Mnih, V., Kavukcuoglu, K.: Multiple object recognition with visual attention. *arXiv pre-print [arXiv:1412.7755](https://arxiv.org/abs/1412.7755)* (2014)
12. Jaderberg, M., Simonyan, K., Zisserman, A.: Spatial transformer networks. In: *Advances in Neural Information Processing Systems*, pp. 2017–2025 (2015)

13. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7132–7141 (2018)
14. Wang, X., Girshick, R., Gupta, A., et al.: Non-local neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7794–7803 (2018)
15. Wang, F., Jiang, M., Qian, C., et al.: Residual attention network for image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2017)
16. Woo, S., Park, J., Lee, J.-Y., Kweon, I.S.: CBAM: convolutional block attention module. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 3–19. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_1
17. Fu, J., Liu, J., Tian, H., et al.: Dual attention network for scene segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3146–3154 (2019)
18. Zhang, N., Donahue, J., Girshick, R.B., et al.: Part-based R-CNNs for fine-grained category detection. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision – ECCV 2014. LNCS. Proceedings, Part I, vol. 8689, pp. 834–849. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_54
19. Xiao, T., Xu, Y., Yang, K., et al.: The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, pp. 842–850. IEEE Computer Society (2015)
20. Fu, J., Zheng, H., Mei, T.: Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In: 2017 IEEE conference on Computer Vision and Pattern Recognition, CVPR, pp. 4476–4484. IEEE Computer Society (2017)
21. Zhang, H., Goodfellow, I., Metaxas, D. and Odena, A. [arXiv:1805.08318](https://arxiv.org/abs/1805.08318)
22. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
23. Zhang, H., et al.: Self-attention generative adversarial networks. In: International Conference on Machine Learning, pp. 7354–7363 (2019)
24. Gao, Z., Xie, J., Wang, Q., et al.: Global second-order pooling convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3024–3033 (2019)
25. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning , pp. 448–456 (2015)
26. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images. Technical Report TR-2009, University of Toronto, Toronto (2009)
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint [arXiv:1409.1556](https://arxiv.org/abs/1409.1556) (2014)
28. Szegedy, C., Liu, W., Jia, Y., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
29. He, K., Zhang, X., Ren, S., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)

30. Huang, G., Liu, Z., Van Der Maaten, L., et al.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
31. Howard, A.G., Zhu, M., Chen, B., et al.: MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint [arXiv:1704.04861](https://arxiv.org/abs/1704.04861) (2017)
32. Szegedy, C., Vanhoucke, V., Ioffe, S., et al.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)



FDRA: Fully Distributed Routing Architecture for Private Virtual Network in Public Cloud

Zhangfeng Hu¹(✉), Hui Zhang¹, Siqing Sun¹, Chuanji Gao¹,
YanJun Li¹, and Xiong Li²

¹ Cloud Computing Research Center, Inspur Cloud Service Group,
Beijing, China

{huzf, zhanghui, sunsq, gaochuanji, i.yanJun}@inspur.com

² School of Computer Science and Engineering, University of Electronic
Science and Technology, Chengdu, China
lixiong@uestc.edu.cn

Abstract. A virtual private cloud (VPC) is often comprised of a set of virtual computing, storage and network resource which is allocated from a public cloud. Public clouds build virtualized resource pools based on physical infrastructure including generic x86 servers, network devices (e.g. switches, routers, firewalls), storage servers and so forth to dynamically provision virtual computing and storage resource to customers, while virtual network is a bridge that connects all the computing resource in a VPC and segregates network traffic between different VPCs. One VPC may contain multiple subnets, which makes private virtual network should provide the capability of communications between virtual computing resources i.e. virtual machines' communication through Layer-2 switching and Layer-3 routing simultaneously. In this paper we propose a fully distributed routing architecture (FDRA) for private virtual network to fulfill the requirements of public cloud. FDRA splits the VPC traffic into two categories, i.e. the traffic goes inside of a VPC and the traffic goes out of a VPC, and presents two different routing entities to route different traffic issued from virtual machines in a VPC.

Keywords: Virtual network · Routing · Fully distributed · Public cloud

1 Introduction

Cloud computing is a distributed computing and storage paradigm, which can provide virtually elastic and almost unlimited scalable service over the Internet for on-demand applications [1]. The rise of public cloud workloads such as Amazon Web Services, Microsoft Azure, Google Cloud Platform, Alibaba Cloud, Inspur Cloud has created a new scale of cloud computing, with vendors regularly reporting server counts in the tens of thousands and even millions [2, 3].

Public cloud is a promising way to provision computing and storage resource to small and medium-sized enterprise customers and even individual customers by severely lowering the threshold of cost and technology in migrating the services running on physical infrastructure to virtual machines which are located in VPCs of a

public cloud [16, 17]. Presently, more and more services are migrating up to the cloud, and this trend should last for a long period of time in the near future. Moving services up to the cloud can bring a lot of benefits, e.g. the capability of dynamically scaling the resources needed by a service based on its current workload, the exemption of building a local data center which often takes a long time to complete the construction, lower CAPEX and OPEX, and so on [4–7]. Technically, the public cloud service providers not only have to provide scale and high performance of virtual machines to customers, but also have to provide rich network functions including private virtual network with separated customer IP address spaces, virtual routing capability between different subnets of a virtual private network, elastic IPs (EIP) to make the virtual machines reachable from outside of the cloud, scalable service load balancers (SLB) to distribute requests to multiple backend real servers for high availability (HA) or performance purpose, security groups and access control lists (ACL) to control the accessibility to virtual machines, and more. Since virtual routing is a basic capability needed for a public cloud, the scalability and performance of virtual routing should be critical to that of a public cloud. Legacy routing architecture is composed of many routers, each router centrally routing the traffic between multiple subnets for a physical network, and dynamic routing protocols running on the routers are used to proliferate the routing information around the whole network [2, 12–18]. This centralized routing architecture has some noticeable defects, e.g., performance bottle neck of routing traffic (because all the traffic between different subnets should go to the centralized router first), scalability limitations (as the physical router cannot be scaled out but scaled up), and some more. This centralized routing architecture works fine for physical network because the scale of the subnets in a network is often pre-planned to be a reasonable and acceptable range, and once the network plan is determined, the maximum number of servers that the network can support is determined. However, for a VPC in a public cloud the scale of the subnets may vary across a wide range e.g. from several virtual machines up to containing tens of thousands of virtual machines, and even cannot be previously determined. The indeterminacy of the subnet scale of VPC makes traditionally centralized routing architecture unsuitable for the routing scenarios in private virtual network of public cloud [21, 22, 26–30].

1.1 Contribution

This paper makes the following contributions:

1. We analyze the design goals and rationale of FDRA.
2. We present the architectural design and implementation of FDRA, and detail the corresponding communication procedure of virtual machines under the architecture.
3. We describe some performance evaluation of FDRA.

1.2 Organization

We introduce the design goals and rationale of private virtual network in Sect. 2, and give the architectural overview of FDRA in Sect. 3. In Sect. 4 we illustrate the design and implementation of FDRA, and in Sect. 5 we analyze the performance of FDRA. We discuss future work in Sect. 6 and conclude the paper in Sect. 7.

2 Design Goals and Rationale

FDRA is a fully distributed routing architecture, which has been evolved over the past several years in Inspur Cloud Service Group. Originally, we designed and implemented a centralized routing architecture for private virtual network in building public cloud and private cloud, which brought some lessons to us, e.g. the scalability problems, the performance issues, lack of robustness, and more.

2.1 Design Goals

Based on the lessons learnt from the centralized routing architecture, we defined the design goals for private virtual network in a public cloud:

1. *Capability of routing traffic between different subnets with high performance (high throughput and low latency).*

More and more Internet services are migrating from physical infrastructure to virtual machines in VPCs of public clouds, resulting in that the private virtual network should provide the capability of connecting virtual machines with high throughput and low latency to facilitate the deployment of all kinds of services. For example, a video service consumes a high throughput of bandwidth, while an online meeting service requires lower latency, and the private virtual network should be adaptable to all the potential services running in VPC.

2. *Segregation of traffic for different virtual private networks.*

A VPC in a public cloud is a separated resource pool of computing (i.e., a set of virtual machines), with a private network connecting all the virtual machines. Different tenants in a public cloud may adopt overlapped IP address space, so we have to segregate the traffic with the same destination IP issued by one tenant from that issued by another.

3. *Good scalability of supporting tens of thousands of virtual machines in a private virtual network.*

Compared to a physical network, the scale of a VPC often cannot be previously planned, and the number of virtual machines running in a VPC may grow up to more than tens of thousands. The private virtual network should be scalable to support the communications between so many virtual machines.

4. *Capability of enabling ACL in routing traffic between different subnets.*

Physical router often provisions the ability to control the accessibility to all the hosts located in a subnet of a network by using ACL, so ACL should also be enabled in a private virtual network to control the resource accessibility.

5. *Support of policy-based routing to fulfill the requirements of some advanced applications.*

Traditional routing behavior is performed based on the destination IP address in the packet. When a packet receives a router, the router looks up in the routing table which is commonly generated by dynamic routing protocols like BGP and OSPF, to find a

routing entry which matches the destination IP address of the packet with the longest prefix to get the next hop information. In some advanced scenarios we have to provide the capability of routing packets based on the source or destination port of transport layer or a combination of source IP, destination IP, transport protocol type, source port of transport layer, destination port of transport layer to direct the traffic flow of a specific service to a specific next hop for further processing, i.e. policy based routing.

2.2 Design Rationale

To make all these goals feasible and implementable we present some design principles before designing the routing architecture:

1. *Separation of control plane and data plane*

Decoupling the control plane and data plane to make both of them evolvable independently.

2. *Software defined virtual networking*

Logically centralized SDN controller defines the forwarding behavior of the data plane. Private virtual network is implemented as a module running in the SDN controller, and the SDN controller translates the definition of a private virtual network into a data plane configuration by which the forwarding behavior of a private virtual network can be realized.

3. *Separation of ID and locator*

ID defines the identity of a virtual machine while locator indicates the current location of a virtual machine. The separation of ID and locator facilitates the live migration of virtual machines, because during the migration only the locator changes while the ID stays unchanged, which guarantees no interruption of communication during the migration.

4. *Separation of east-west traffic and north-south traffic*

Splitting the traffic that going out of a subnet into two parts, the traffic going to another subnet of the VPC and the traffic going out of the VPC. The traffic of former part is often very high and needs high performance routing capability, while the traffic of latter part is not that huge which needs rich and flexible processing, e.g. NAT (network address translation), bandwidth metering, and ACL application.

5. *Combination of distributed east-west routing entities and centralized north-south routing entities*

Distributed virtual routers which can support high forwarding throughput and good scalability are needed to efficiently route the traffic issued in a VPC between different subnets, but centralized virtual routers which provides rich and flexible network functions are needed to flexibly route the traffic going out to Internet through NAT, rate limiting, or ACL.

3 Architectural Overview

Figure 1 gives the architectural overview of the network component used in Inspur public cloud. Public cloud orchestration layer is responsible for orchestrating all the resources virtualized in a public cloud, and provides cloud services and products as ECS (elastic cloud server, i.e. virtual machine), EIP (elastic IP), SLB (service load balancer), and more to customers.

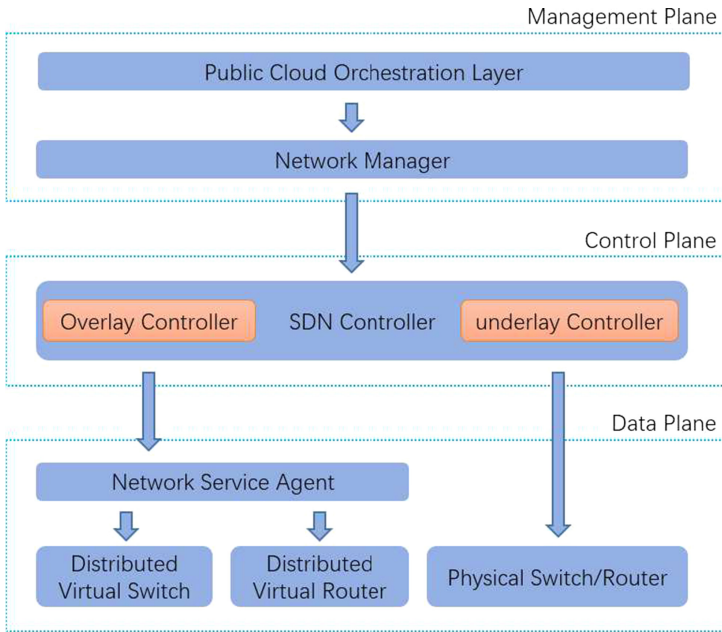


Fig. 1. Architecture of network component in Inspur public cloud.

Network manager builds virtual private networks for VPC on top of a series of physical network equipment, including switches, routers, firewalls, and so on. SDN controller consists of two sub-modules, i.e. underlay controller and overlay controller. Underlay controller is responsible for controlling the configuration and forwarding behavior of the underlay network, generally the switches and the routers, while the overlay controller is responsible for the management of virtual private networks and configures the network service agents located in the data plane to support networking connectivity for virtual private networks. Network service agents translate the semantics of virtual private networks into specific configurations of distributed virtual switches and routers in the data plane, for example some flow entries in a virtual switch or routing entries in a virtual router. The distributed virtual switches construct a huge switching fabric to connect all the virtual machines while segregating the traffic of different tenants by using VLAN and VXLAN.

4 A Fully Distributed Routing Architecture

As mentioned above we have presented the architectural overview of the framework of Inspur public cloud, and in this section we will illustrate the details of the routing scheme for virtual private networks in the public cloud.

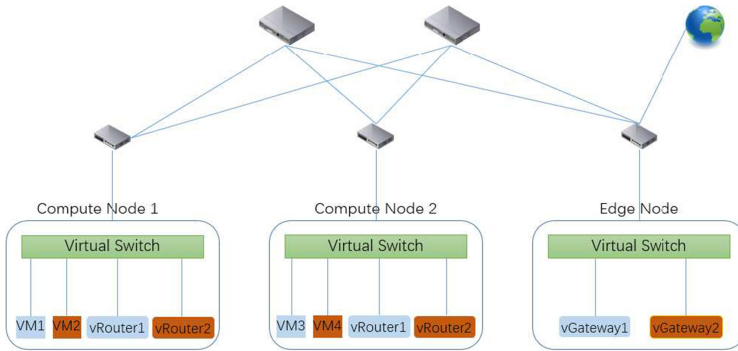


Fig. 2. A fully distributed routing architecture.

4.1 Distributed Virtual Routers

As is shown in Fig. 2 we distribute the routers to be implemented on all the compute nodes that are concerned with the virtual machines running on them. One virtual private network (VPC) has only one virtual router from the users’ perspective. But the only router may have multiple distributed instances, and these instances share the same IP addresses and the same MAC addresses of the default gateways of the subnets included, as is shown in Fig. 3.

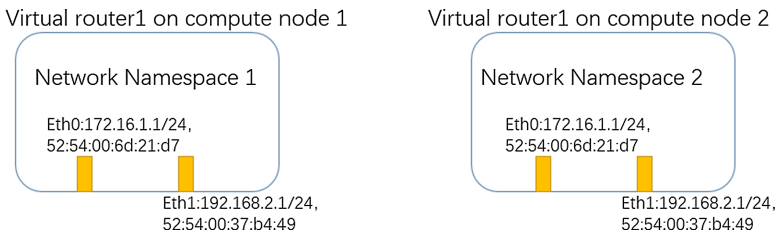


Fig. 3. Inner implementation of a distributed router.

We implement the distributed virtual routers by using the network namespace of Linux system. When a router is created, the network service agent creates a network namespace, and when a subnet is created in a virtual private network a pair of Linux VETH (Virtual Ethernet Device) ports are created with one side being in the network namespace and the other side connecting the virtual switch.

4.2 Routing Traffic Going Inside of VPC

Routing traffic under FDRA is different from traditional routing methods, and we will illustrate the routing method under FDRA with an example. Suppose two tenants in the public cloud, and each tenant has a VPC of two subnets, vm1 and vm3 in subnets of tenant 1, vm2 and vm4 in subnets of tenant 2 individually, as Fig. 2 shows. When vm1 located on compute node 1 issues a communication to vm3 located on compute node 2, the steps are as follows:

1. Vm1 issues an ARP request which is broadcast to the whole private virtual network to find the mac address of the default gateway of the subnet it belongs to.
2. vRouter1 on compute node 1 receives the ARP request and replies the MAC address of the default gateway of the subnets. (Please note: As the virtual switch prohibits all the ARP requests to the subnet gateways from going out of the compute node, only the virtual router located on the same compute node can reply the ARP request.)
3. Vm1 send the first packet (e.g. TCP SYN packet) destined to vm3 to vRouter1 on compute node1.
4. vRouter1 forwards the packet to vm3 by using VXLAN encapsulation. As the virtual router keeps the mappings of IP address to MAC address for all virtual machines in a VPC, vRouter1 does not need to request the MAC of vm3 through ARP.
5. Vm3 replies a packet (e.g. TCP SYN&ACK packet) to vm1.
6. Vm3 issues an ARP request which is broadcast to the whole private virtual network to find the mac address of the default gateway of the subnet it belongs to.
7. vRouter1 on compute node 2 receives the ARP request and replies the MAC address of the default gateway of the subnets.
8. vRouter1 forwards the packet to vm1 by using VXLAN encapsulation. As the virtual router keeps the mappings of IP address to MAC address for all virtual machines in a VPC, vRouter1 does not need to request the MAC of vm1 through ARP.

As the procedure illustrated above, each virtual machine uses the local virtual router located on the same compute node to route traffic to different subnets. Each virtual router keeps the IP-to-MAC mappings for all the virtual machines in the VPC, so the virtual routers do not need to get the MAC address of a virtual machine through ARP. By default, each virtual switch on the compute node injects a flow entry into the flow table to drop the ARP requests going out of the compute node, which avoids multiple ARP replies to a same ARP request.

4.3 Routing Traffic Going Out of VPC

Traffic going out of VPC is in some degree different from that going inside of VPC. When a packet goes out of a VPC, SNAT (Source Network Address Translation) should be executed to replace the source IP address of the packet with a globally routable IP address (i.e. EIP) before it leaves the VPC. We deploy a centralized virtual gateway for each VPC to route the north-south traffic to the Internet. As the north-south

traffic consumes WAN (Wide Area Network) link resource, the traffic bandwidth of north-south is often limited unlike the bandwidth of east-west traffic. We use an example to illustrate the procedure of visiting an Internet service from a virtual machine inside of a VPC. As Fig. 2 shows, vm1 visits an Internet website of IP address being 104.193.88.77, the steps are as follows:

1. Vm1 issues an ARP request which is broadcast to the whole private virtual network to find the mac address of the default gateway of the subnet it belongs to.
2. vRouter1 on compute node 1 receives the ARP request and replies the MAC address of the default gateway of the subnets.
3. Vm1 send the first packet (e.g. TCP SYN packet) destined to 104.193.88.77 to vRouter1 on compute node1.
4. vRouter1 forwards the packet to vGateway1 which is located on edge node by using VXLAN encapsulation.
5. vGateway1 performs SNAT to replaces the source IP address of the packet with a globally routable IP address and then sends it out from the interface which is connected to the external network. Since the IP address used by a VPC is privately and locally routable only inside of a VPC, an EIP must be bound to a virtual machine in order to make it reachable from the Internet.
6. The packet is then routed by the routers in the Internet hop by hop until it arrives the destination.
7. When the response comes back from 104.193.88.77, it is also routed by the routers in the Internet hop by hop until it arrives vGateway1 located on the edge node.
8. vGateway1 performs DNAT to replace the destination IP address of the packet with the IP address of vm1, and sends the packet to vm1 by using VXLAN encapsulation.

A centralized virtual gateway is used to perform SNAT and DNAT for the traffic going out and from the Internet for the virtual machines in a VPC. As the north-south traffic needs the support of WAN links, the north-south traffic volume is not so huge as that of east-west traffic. In our design each, one edge node can support up to 4X40G (160G) bandwidth for north-south traffic, and multiple edge nodes can be deployed to support higher bandwidth.

5 Performance Analysis

Now we analyze the performance of FDRA compared to traditional centralized routing architecture.

5.1 Communication Delay

Under centralized routing architecture, the delay of transmitting a packet from source to destination can be expressed as Eq. (1)

$$T_{delay1} = T_{transmit} + T_{sender_to_router} + T_{forwarding} + T_{router_to_recipient} + T_{receive} \quad (1)$$

While for FDRA, the delay of transmitting a packet from source to destination can be expressed as Eq. (2)

$$T_{delay2} = T_{transmit} + T_{forwarding} + T_{source_to_destination} + T_{receive} \quad (2)$$

As is known,

$$T_{source_to_destination} < T_{sender_to_router} + T_{router_to_recipient} \quad (3)$$

So, we can get the conclusion as Eq. (4), which indicates the delay under FDRA is lower than that under traditional centralized routing architecture.

$$T_{delay2} < T_{delay1} \quad (4)$$

Figure 4 gives the experimental results which prove the correctness of Eq. (4). Under FDRA, we can get a lower communication delay when establishing a communication session inside of a VPC.

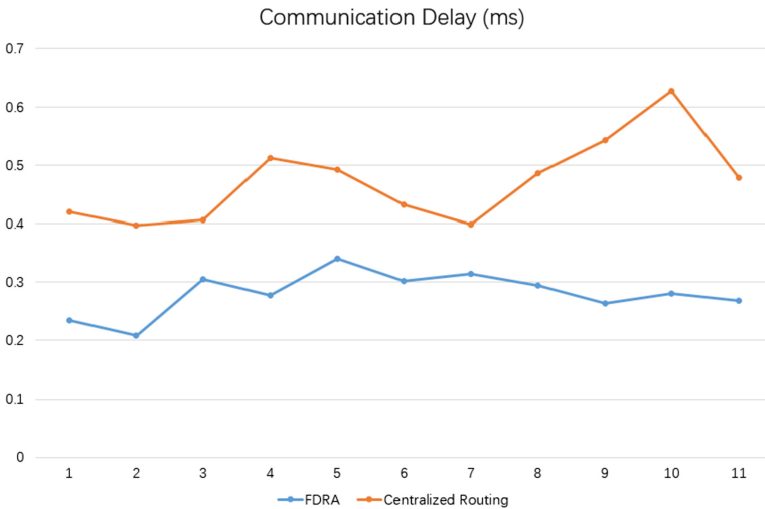


Fig. 4. Experimental results of communication delay.

5.2 Throughput

Routing throughput under traditional centralized architecture is decided by the performance of the centralized virtual router. If we deploy a virtual router with the capability of supporting 5 Gigabit of routing traffic per second, then the routing throughput of a VPC is limited to 5Gbps. But in FDRA, the virtual router is fully

distributed and deployed on every compute node concerned, so the routing throughput is expressed as Eq. (5)

$$P_{total} = N_{concerned_nodes} * P_{vRouter} \quad (5)$$

Under FDRA the routing throughput is decided by the performance of virtual router and also the number of compute nodes that are concerned with a specific VPC. The more compute nodes are concerned, the higher routing performance a VPC can obtain. Figure 5 gives the experimental results of routing throughput under FDRA and also centralized routing architecture, each testbed including four compute nodes.

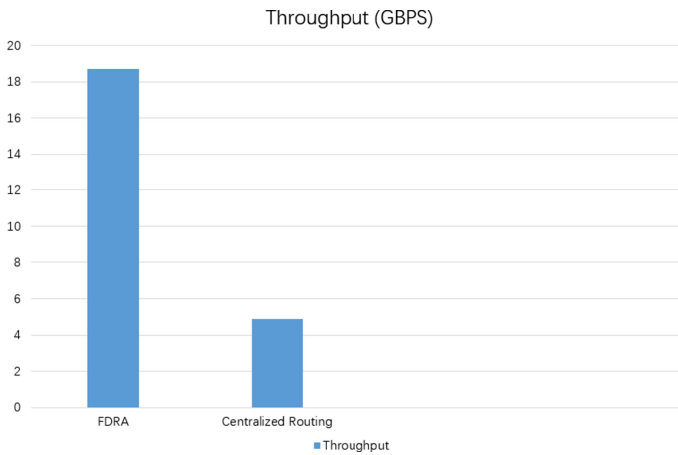


Fig. 5. Experimental results of routing throughput.

6 Future Work

In future work, we aim to adopt DPDK into the implementation of virtual router and virtual gateway, and design a clustered virtual gateway to provide higher performance to process north-south traffic for a single VPC. And we will design a clustered load balancer scheme under FDMA.

7 Conclusion

Traditionally centralized routing architecture cannot fulfill the requirements of public cloud because it has some obvious defects as lack of scalability, performance bottle neck, lack of robustness, triangular routing path which cause higher communication delay, and more. In this paper we present a fully distributed routing architecture for private virtual network, which has been implemented in Inspur public cloud. Under FDRA, virtual routers are deployed in distributed mode, and all the traffic going across subnets are routed by the local virtual router, and the traffic going out of a VPC

is routed to a centralized virtual gateway for further process, e.g. NAT, rate limiting, ACL checking. We evaluate the routing performance of RDMA, and the results show that it has lower communication delay and higher routing throughput under general scenarios.

Acknowledgement. This work is supported by the National Natural Science Foundation of China under Grant no. 62072078, and the Natural Science Foundation of Hunan Province, China under Grant no. 2018JJ3191.

References

1. Paladi, N., Gehrmann, C.: TruSDN: bootstrapping trust in cloud network infrastructure. In: Deng, R., Weng, J., Ren, K., Yegneswaran, V. (eds.) *SecureComm 2016*. LNICSSITE, vol. 198, pp. 104–124. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-59608-2_6
2. Yang, S., Li, F., Trajanovski, S., Chen, X., Wang, Y., Fu, X.: Delay-aware virtual network function placement and routing in edge clouds. *IEEE Trans. Mobile Comput.* (2019)
3. Firestone, D.: VFP: a virtual switch platform for host SDN in the public cloud. In: *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2017)*, pp. 315–328 (2017)
4. Hu, B., Chen, S., Chen, J., Hu, Z.: A mobility-oriented scheme for virtual machine migration in cloud data center network. *IEEE Access.* **17**(4), 8327–8337 (2016)
5. Otokura, M., Leibnitz, K., Koizumi, Y., Kominami, D., Shimokawa, T., Murata, M.: Evolvable virtual network function placement method: mechanism and performance evaluation. *IEEE Trans. Netw. Serv. Manage.* **16**(1), 27–40 (2019)
6. Kwon, J., Lee, T., Hähni, C., Perrig, A.: SVLAN: secure & scalable network virtualization. In: *Proceedings of the Symposium on Network and Distributed System Security (NDSS) (2020)*
7. Xiao, X., Zheng, X., Zhang, Y.: A multidomain survivable virtual network mapping algorithm. *Secur. Commun. Netw.* (2017)
8. Martini, B., Paganelli, F.: A service-oriented approach for dynamic chaining of virtual network functions over multi-provider software-defined networks. *Future Internet.* **8**(2), 24 (2016)
9. Miotto, G., Luizelli, M.C., da Costa Cordeiro, W.L., Gaspary, L.P.: Adaptive placement & chaining of virtual network functions with NFV-PEAR. *J. Internet Serv. Appl.* **10**(1), 3 (2019)
10. Dwaraki, A., Wolf, T.: Adaptive service-chain routing for virtual network functions in software-defined networks. In: *Proceedings of the 2016 Workshop on Hot Topics in Middleboxes and Network Function Virtualization, 22 August 2016*, pp. 32–37 (2016)
11. Sauvanaud, C., Lazri, K., Kaâniche, M., Kanoun, K.: Anomaly detection and root cause localization in virtual network functions. In: *2016 IEEE 27th International Symposium on Software Reliability Engineering (ISSRE), 23 October 2016*, pp. 196–206. IEEE (2016)
12. Carpio, F., Jukan, A., Pries, R.: Balancing the migration of virtual network functions with replications in data centers. In: *NOMS 2018–2018 IEEE/IFIP Network Operations and Management Symposium, 23 April 2018*, pp. 1–8. IEEE (2018)
13. Kim, D., Kim, Y.H., Kim, K.H., Gil, J.M.: Cloud-centric and logically isolated virtual network environment based on software-defined wide area network. *Sustainability* **9**(12), 2382 (2017)

14. Cao, H., Zhu, H., Yang, L.: Collaborative attributes and resources for single-stage virtual network mapping in network virtualization. *J. Commun. Netw.* **22**(1), 61–71 (2019)
15. Moreno-Vozmediano, R., Montero, R.S., Huedo, E., Llorente, I.M.: Cross-site virtual network in cloud and fog computing. *IEEE Cloud Comput.* **4**(2), 46–53 (2017)
16. Su, Y., Meng, X., Kang, Q., Han, X.: Dynamic virtual network reconfiguration method for hybrid multiple failures based on weighted relative entropy. *Entropy* **20**(9), 711 (2018)
17. Liu, X., Medhi, D.: Dynamic virtual network restoration with optimal standby virtual router selection. In: *NOMS 2016–2016 IEEE/IFIP Network Operations and Management Symposium*, 25 April 2016, pp. 973–978. IEEE (2016)
18. Maswood, M.M., Develder, C., Madeira, E., Medhi, D.: Dynamic virtual network traffic engineering with energy efficiency in multi-location data center networks. In: *2016 28th International Teletraffic Congress (ITC 28)*, 12 September 2016, vol. 1, pp. 10–17. IEEE (2016)
19. Chandramouli, R., Chandramouli, R.: Secure virtual network configuration for virtual machine (VM) protection. *NIST Spec. Publ.* **7**(800), 125B (2016)
20. Alaluna, M., Ferrolho, L., Figueira, J.R., Neves, N., Ramos, F.M.: Secure virtual network embedding in a multi-cloud environment. arXiv preprint [arXiv:1703.01313](https://arxiv.org/abs/1703.01313), March 2017
21. Fischer, A., De Meer, H.: Generating virtual network embedding problems with guaranteed solutions. *IEEE Trans. Netw. Serv. Manage.* **13**(3), 504–517 (2016)
22. Pei, J., Hong, P., Xue, K., Li, D.: Efficiently embedding service function chains with dynamic virtual network function placement in geo-distributed cloud system. *IEEE Trans. Parallel Distrib. Syst.* **30**(10), 2179–2192 (2018)
23. Alzahrani, A.S., Shahin, A.A.: Energy-aware virtual network embedding approach for distributed cloud. arXiv preprint [arXiv:1710.11590](https://arxiv.org/abs/1710.11590), 31 October 2017
24. Maswood, M.M., Develder, C., Madeira, E., Medhi, D.: Energy-efficient dynamic virtual network traffic engineering for north-south traffic in multi-location data center networks. *Comput. Netw.* **9**(125), 90–102 (2017)
25. Comi, P., et al.: Hardware-accelerated high-resolution video coding in virtual network functions. In: *2016 European Conference on Networks and Communications (EuCNC)*, 27 June 2016, pp. 32–36. IEEE (2016)
26. Savi, M., Tornatore, M., Verticale, G.: Impact of processing-resource sharing on the placement of chained virtual network functions. *IEEE Trans. Cloud Comput.* (2019)
27. Dräxler, S., Karl, H., Mann, Z.Á.: Jasper: joint optimization of scaling, placement, and routing of virtual network services. *IEEE Trans. Netw. Serv. Manage.* **15**(3), 946–960 (2018)
28. AbdelSalam, A., Clad, F., Filsfils, C., Salsano, S., Siracusano, G., Veltri, L.: Implementation of virtual network function chaining through segment routing in a Linux-based NFV infrastructure. In: *2017 IEEE Conference on Network Softwarization (NetSoft)* 3 July 2017, pp. 1–5. IEEE (2017)
29. Li, H., Ota, K., Dong, M.: LS-SDV: virtual network management in large-scale software-defined IoT. *IEEE J. Sel. Areas Commun.* **37**(8), 1783–1793 (2019)
30. Arouk, O., Nikaiein, N., Turletti, T.: Multi-objective placement of virtual network function chains in 5G. In: *2017 IEEE 6th International Conference on Cloud Networking (CloudNet)*, 25 September 2017, pp. 1–6. IEEE (2017)
31. Esposito, F., Di Paola, D., Matta, I.: On distributed virtual network embedding with guarantees. *IEEE/ACM Trans. Networking* **24**(1), 569–582 (2014)
32. Gupta, A., Habib, M.F., Mandal, U., Chowdhury, P., Tornatore, M., Mukherjee, B.: On service-chaining strategies using virtual network functions in operator networks. *Comput. Netw.* **14**(133), 1–6 (2018)
33. Gao, M., Addis, B., Bouet, M., Secci, S.: Optimal orchestration of virtual network functions. *Comput. Netw.* **4**(142), 108–127 (2018)



Energy Consumption Modeling for the PageRank Application in Spark

Yunlan Wang^(✉), Yingjing Wang, Zhengxiong Hou, and Xingli Li

School of Computer Science, Northwestern Polytechnical University,
Xi'an, Shaanxi, China
wangyl@nwpu.edu.cn

Abstract. Energy consumption caused by processing massive data is growing explosively with the rapid growth of big data applications. PageRank is one of the typical algorithms that is frequently used by Google when evaluating the importance of web pages in search engine optimization. In this paper, an energy consumption model is proposed for the PageRank application to reduce energy consumption. We analyze the influence of some important features on the power consumption and run time of the PageRank application, including CPU frequency, Spark configuration parameters, and application characteristics such as the number of pages and iterations. Linear regression and multivariate polynomials regression are adopted as the main modeling algorithms. The experimental results show that the accuracy of energy consumption of PageRank applications can reach 87% in large-scale data. Based on the proposed energy consumption model, we can adjust the CPU frequency, the number of tasks and the number of computer nodes to optimize energy consumption.

Keywords: Big data · PageRank · Energy Consumption model · Energy optimization

1 Introduction

Data centers are critical, energy-hungry infrastructures that run large-scale Internet-based services. The huge energy consumption caused by processing data has become one of the problems to be solved urgently. It will not only increase the operating cost of the computing and storage system, but also bring more pressure to our ecological environment. This problem will become more serious with the further development of big data and its application. It is imperative to optimize the energy consumption of big data processing while ensuring the computing performance requirement of applications.

The first step to reduce energy consumption when processing massive data is monitoring the energy consumption of big data applications. IPMI (Intelligent Platform Management Interface) is an open-standard hardware management interface specification which can be used to monitor the physical health characteristics of the server, such as temperature, voltage, power status, and so on [1]. IPMI can be applied to different operating systems, firmware and hardware platforms, control and automatically report the operation status of a large number of servers, to reduce the cost of the server system.

Big data has penetrated almost every aspect of our lives. Apache Spark, a fast and general compute engine that is designed for large-scale data processing has become the most popular open-source big data processing platform at present [2]. Spark is compatible with Hadoop ecosystem and can replace MapReduce computing framework to a certain extent. Spark inherits many advantages of MapReduce and overcomes the shortcomings of MapReduce. In the foreseeable future, Spark will still be the cornerstone of big data analysis and the key research content of scientific and technological.

A variety of applications run in the data center. In this paper, we focus on the PageRank application. PageRank is an algorithm proposed by Larry Page and Sergey Brin in 1998 to evaluate the importance of web pages. The application of the PageRank algorithm makes Google a great success, and sets off the climax of network weblink analysis. Our workload comes from HiBench, which was originally developed by Intel, and is now open source for the Apache community. As a benchmark of big data, HiBench can help evaluate the performance of different big data frameworks in terms of running speed, throughput and system resource utilization.

In this paper, we propose an energy consumption model, which can fit well the practical energy consumption of the PageRank application running on the cluster under a specific data scale. Based on the model, we can minimize energy consumption by setting Spark parameters or adjusting the CPU frequency.

The rest of the paper is organized as follows: in the Sect. 2, we introduce the existing related research work. In the Sect. 3, we describe the energy consumption model of the PageRank application. In the Sect. 4, we show the experimental results of the model, including minimizing energy consumption by adjusting Spark parameters or CPU frequency. Finally, we summarize this paper and introduce some future work.

2 Related Work

Data center energy consumption management has always been a research hotspot in the big data field. Generally speaking, an effective energy consumption management technology needs to consider three key factors: firstly, the data center get energy consumption data through energy monitoring systems; secondly, through analysis and mathematical modeling, the relationship model about energy consumption and system state can be established, which can accurately predict energy consumption; finally, energy efficiency optimization algorithms or energy-saving technologies are designed and implemented to meet the performance and QoS (Quality of Service) requirements while reducing the energy consumption of the data center.

At present, many scholars pay attention to the energy consumption of cloud data centers. Lim et al. clearly pointed out that energy efficiency has become an important consideration in the design of Internet data center [3]. Under the condition of Service Level Agreement (SLA), the energy consumption of data centers can be reduced through dynamic server configuration, frequency regulation and dynamic power management. Masdari et al. provided a complete survey of the existing state of the art VM placement schemes proposed in the literature for the cloud computing and data centers [4]. Mann et al. focused on the classification and different research of current virtual machine deployment research [5]. Through the study of the above literature, it is

found that energy consumption and quality of service (QoS) are the most noteworthy issues.

The energy consumption modeling is the basis of energy consumption optimization. Many scholars have conducted in-depth research on the energy consumption model of cloud computing data centers. Luo et al. analyzed and compared a variety of mathematical methods, using the performance counter and system utilization of the processor as parameters to model server energy consumption [6]. Dayarathna et al. studied and summarized more than 200 existing energy consumption models of data centers from both hardware and software aspects. This systematic method allows us to identify multiple problems in energy consumption models of different levels of data center [7].

In order to effectively reduce the energy consumption of the data center and maintain good computing performance, many scholars have also done some research on the energy consumption of big data applications. Wu et al. reviewed the studies on improving energy efficiency of Hadoop clusters and summarized them in five categories. For each category, they briefly illustrate its rationale and comparatively analyze the relevant works regarding their advantages and limitations [8]. Mashayekhy et al. proposed a framework to improve the energy efficiency of MapReduce applications while meeting the SLA (Service Level Agreement) [9]. Li et al. presented a new energy consumption model based on Spark framework, then proposed an energy-aware scheduling algorithm EASAS for Spark to reduce energy consumption [10]. However, the energy consumption models proposed above are all general models based on a big data framework, and there is no energy consumption model for specific big data applications.

3 Energy Consumption Model of PageRank Application

Energy consumption model plays a fundamental role in energy-efficiency research. In this section, we analyze the power consumption model, runtime model and energy consumption model in turn. Table 1 summarizes all the parameters used in this paper.

3.1 Power Consumption Model

Many server power models are presented in which system resource utilization are considered. Traditionally, CPU power consumption accounts for a large part of server power consumption. Therefore, most power models based on system utilization take CPU utilization as the primary index to model the whole system power consumption. According to reference [11], the relationship between power consumption and CPU frequency is a quadratic function. Through the experiment, we found that the power consumption increases with the increase of data size, so we take CPU frequency f and page number $page$ as the variables of power consumption model.

When we use multiple linear regression model for analysis, the fitting effect is not so satisfactory, because we simply think that the power consumption P is linear with frequency f and page number $page$, which will distort the internal relationship between

data and make the prediction model inaccurate. Therefore, we consider introducing a nonlinear regression model using multivariate polynomials.

$$P = \theta_0 + \theta_1 f + \theta_2 page + \theta_3 f^2 + \theta_4 fpage + \theta_5 page^2 + \Lambda \quad (1)$$

θ_i is regression coefficient. Without knowing the most suitable number of polynomials, we start from the quadratic, gradually increase the degree and check the error size, and finally obtain the appropriate high order term. At the same time, we should pay attention to prevent overfitting problem.

Table 1. Summary of model parameters

Variable	Description
n	The number of data
$page$	The number of pages
t	The number of tasks in each stage
w	The number of worker nodes in the cluster
k	The number of iterations of PageRank algorithm
mem	memory size
f	CPU frequency
t/w	The number of tasks processed by each worker node
$page/t$	The number of pages processed by each task
n/t	The number of data processed by each task
T_{ave}	Average running time of $Stage_3$ to $Stage_{k+1}$
T_{esum}	The total running time of $Stage_0$, $Stage_1$, $Stage_2$ and $Stage_{k+2}$
T	Running time of PageRank application
P	Average power consumption of worker nodes when PageRank application is running
E	Energy consumption of running PageRank application

3.2 Runtime Model

Suppose that the PageRank application has n data, $page$ pages, and k iterations. In default standalone model, there is one executor process running on a worker node. Suppose we set w worker nodes, so there are w executor processes. PageRank application has only one action operator, `saveAsTextfile`, so it consists of one job. According to the dependency relationship between RDDS, a job can be composed of one or more stages. A stage consists of one or more tasks. In the PageRank application, each stage has the same number of tasks, and we can modify the HiBench configuration file to change the number of tasks per stage. Assuming that the number of tasks in each stage is t .

PageRank is an algorithm that performs multiple iterations, which mainly maintains two datasets, links and ranks. Links are composed of (pageId, linkList) elements and contain a list of adjacent pages of each page. Ranks are composed of (pageId, rank) elements and contain the current rank value of each page. From the DAG logic diagram, it can be seen that PageRank workload consists of one Job which contains $k + 3$ stages $Stage_0, Stage_1, \dots, Stage_{k+2}$, where k is the number of iterations in the algorithm.

According to DAG logic diagram, $Stage_0$ mainly reads data from HDFS to form the set of ($pageId_1, pageId_2$), where $pageId_1$ is a page and $pageId_2$ is a linked page of $pageId_1$. $Stage_1$ uses distinct operator to remove duplicate elements. $Stage_2$ first uses groupByKey operator to form the dataset links, and assign the initial value of ranks, that is, the rank value of all pages is set to 1. Then links and ranks are obtained through join and map operators to get the rank value of each page contributing to its linked out page. The operations from $Stage_3$ to $Stage_{k+1}$ are the same. First, add the rank values with the same pageId in the data obtained from $Stage_2$ to get the original ranks of the page, then use the map operator to compute $0.15 + 0.85 * originalranks$ to form new ranks, and then connect the new ranks with links through join operator, finally perform the same operation as $Stage_2$. $Stage_{k+2}$ uses the reduceByKey operator to get the final rank value of all pages, and writes the results to the file.

Since the operations from $Stage_3$ to $Stage_{k+1}$ are the same, and the running time of this $k - 1$ stage is similar according to the experiment, we use T_{ave} to represent the average running time of the $k - 1$ stage, and T_{esum} to represent the total running time of the other four stages, $Stage_0, Stage_1, Stage_2$, and $Stage_{k+2}$. Therefore, the total running time of the PageRank application can be expressed as follows:

$$T = (k - 1)T_{ave} + T_{esum} \quad (2)$$

Traditionally, we think that the running time of a program is negatively correlated with the CPU frequency f , that is, the higher the f , the shorter the program execution time. As a distributed computing engine based on memory, the memory management module of Spark plays a very important role in the whole system. By adjusting the memory size in the HiBench configuration file, we find that the running time decreases with the increase of memory. According to the related concepts of Spark and the related theories mentioned above, the running time of parallel programs is also related to the number of tasks, the number of nodes and the amount of data, so we built the T_{ave} and T_{esum} models as follows.

$$T_{ave} = \left(\frac{t}{w}\right)^{c_1} \left(\frac{page}{t}\right)^{c_2} mem^{c_3} * f^{c_4} \quad (3)$$

$$T_{esum} = \left(\frac{t}{w}\right)^{c_5} \left(\frac{n}{t}\right)^{c_6} mem^{c_7} * f^{c_8} \quad (4)$$

Where f is CPU frequency, unit is MHz, mem is memory size, unit is GB, $\frac{t}{w}$ is the number of tasks run by each worker node, $\frac{page}{t}$ and $\frac{n}{t}$ are the number of pages and data

processed by each task respectively. c_i is a indeterminate constant. Based on (2), (3), and (4), the PageRank application runtime T can be expressed as follows.

$$T = (k - 1) * \left(\frac{t}{w}\right)^{c_1} \left(\frac{m}{t}\right)^{c_2} mem^{c_3} * f^{c_4} + \left(\frac{t}{w}\right)^{c_5} \left(\frac{n}{t}\right)^{c_6} mem^{c_7} * f^{c_8} \quad (5)$$

3.3 Energy Consumption Model

We use E to present the energy of the PageRank application over a time period T , while P is the power in the above mentioned time period. The relationship of E , P and T can be expressed by formula (6).

$$E = P * T \quad (6)$$

Since we assume that the worker nodes in the computer cluster which run the PageRank application are homogeneous, and each node does the same amount of work, the total energy consumption can be expressed as formula (7).

$$E = w * P * T \quad (7)$$

Where E is the system's energy consumption measured in Joules, w represents the number of worker nodes, P is the power measured in Watts and T is a period of time measured in seconds. If T is measured in unit times and only one worker node then the values of energy and power become equal.

4 Experimental Results

Our experiment platform has four computing nodes in a multi-core cluster. Each node has four Intel(R) Xeon(R) Gold 6132 fourteen-core processors. The maximum frequency is 2.6 GHz, the minimum frequency is 1.0 GHz. Each node has 376 GB memory. The PageRank application used in the experiment comes from HiBench. We write a monitoring script file which uses the IPMI interface to obtain the power consumption of the worker node. The running time of each stage of the PageRank application can be obtained from Spark's log file.

4.1 Results of Power Consumption Model

We use the Curve Fitting Tool of MATLAB to do polynomial fitting. RMSE and R-square are used to represent the error and fitting effects of the model. RMSE is the root of mean square error, that is, the square root of the mean square of the sum of squares of the errors corresponding to the predicted data and the measured data. The R-square represents the fitting degree of the model to the data. Its value range is [0,1] and the closer the value is to 1, the better the model is. We evaluate the model errors with different degrees, which are shown in Tables 2.

Considering that the model can predict power consumption better and the complexity is not too high, we set the degree of f and page to 2. Therefore, the power consumption model is as follows.

Table 2. Results of multivariate polynomial model.

Degree of f	Degree of page	RMSE	R-square
2	1	13.95	0.8867
1	2	13.21	0.8984
2	2	13.11	0.9024
3	1	13.83	0.8943
3	2	12.92	0.9125
1	3	11.3	0.9294
2	3	11.16	0.9202
3	3	11.31	0.9348

Table 3. Parameter information of power consumption model.

θ_0	θ_1	θ_2	θ_3	θ_4	θ_5
-54.95	0.2503	3.024e-06	-2.997e-05	-5.999e-11	-5.878e-14

$$P = \theta_0 + \theta_1 f + \theta_2 \text{page} + \theta_3 f^2 + \theta_4 f \text{page} + \theta_5 \text{page}^2 \quad (8)$$

We use MATLAB to obtain the value of θ_i by parameter fitting, the results are shown in Table 3.

As shown in Fig. 1, the abscissa is the identifier of experiments, and the ordinate is the power consumption value. The blue line represents the predicted value and the orange line represents the measured value of the operation. It can be seen that polynomial model can fit the data. The RMSE of this model is 13.11 and the R-square is 0.9024.

4.2 Results of Runtime Model

According to the description in Sect. 3.2, we divide the PageRank application runtime model into two parts: T_{ave} and T_{esum} . In the experiment, the parameters of the two parts are fitted and the results are compared. Finally, we combine the two parts according to formula (5) to analyze the accuracy of the total runtime model.

Parameters c_i of T_{ave} and T_{esum} models are fitted by 1stop software, and the results are shown in Table 4 and Table 5. The parameters and experimental data are substituted into the two models to compare the predicted value with the actual value, and the results are shown in Fig. 2 and Fig. 3. The RMSE of T_{ave} and T_{esum} models are 7.734 and 25.816, and R-square are 0.949 and 0.972. We put the data from the above time models into formula (5) to get the predicted total running time of the PageRank

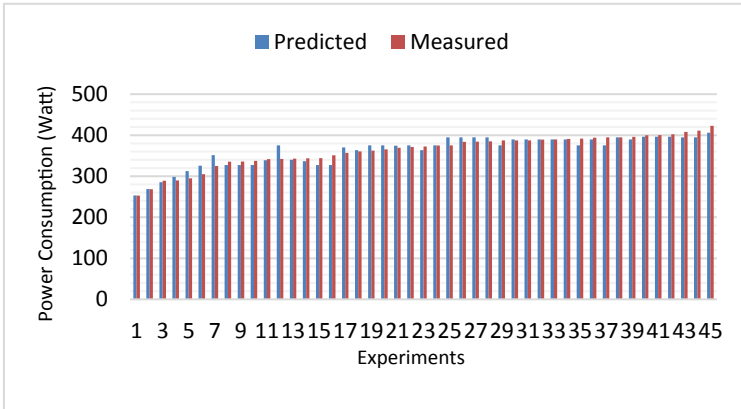


Fig. 1. Distribution of predicted and measured values of power consumption.

application, and compare it with the actual measured running time. The results are shown in Fig. 4.

Table 4. Parameter information of T_ave model.

c_1	c_2	c_3	c_4
1.453	1.642	-0.909	-2.141

As shown in Fig. 3, Fig. 4, and Fig. 5, the abscissa is the identifier of experiments, and the ordinate is the running time in seconds. The blue line represents the predicted value and the orange line represents the measured value of the operation.

4.3 Results of Energy Consumption Model

According to formula (7), energy consumption E can be expressed by power P , time T and number of nodes w . We multiply the number of nodes w with the data predicted by the power consumption model and the runtime model to obtain the energy consumption data, and compare it with the actual measured energy consumption data. The results are shown in Fig. 5.

As shown in Fig. 5, the abscissa is the identifier of experiments, and the ordinate is the energy consumption in joules. The blue line represents the predicted value and the orange line represents the measured value of the operation. Finally, the experimental data show that our model is suitable for predicting the energy consumption of large data sets, and the average accuracy rate can reach 87%.

Table 5. Parameter information of T_esum model.

c_5	c_6	c_7	c_8
0.738	0.945	-0.175	-1.611

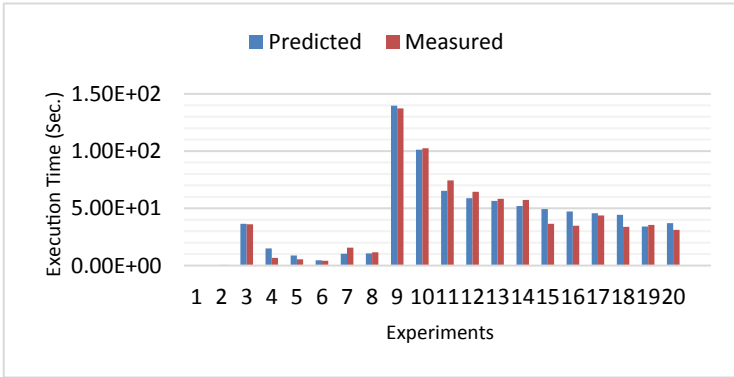


Fig. 2. Distribution of predicted and measured values of T_ave.

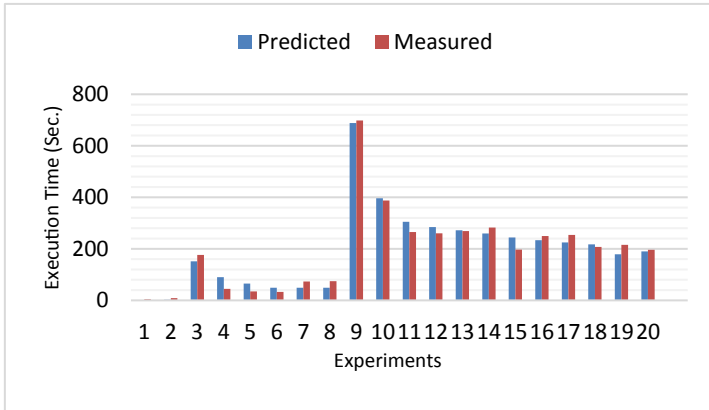


Fig. 3. Distribution of predicted and measured values of T_esum.

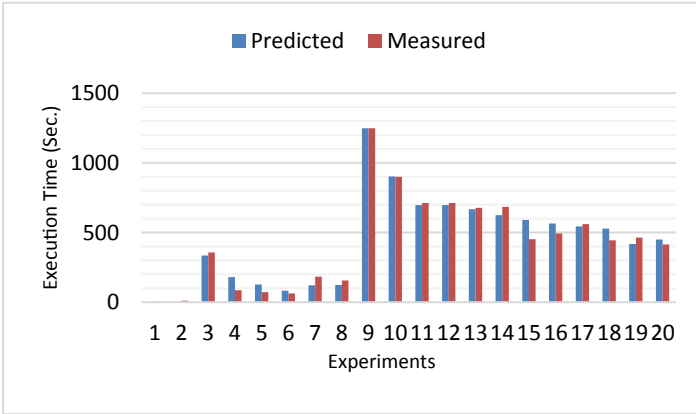


Fig. 4. Distribution of predicted and measured values of runtime.

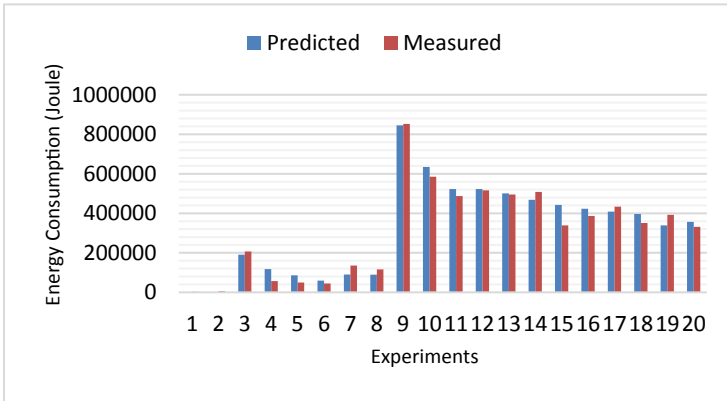


Fig. 5. Distribution of predicted and measured values of energy consumption.

4.4 Energy Consumption Optimization

According to formula (1)–(8), we can get the final energy consumption model as follows.

$$E = w * (\theta_0 + \theta_1 f + \theta_2 page + \theta_3 f^2 + \theta_4 f page + \theta_5 page^2) * \left[(k - 1) * \left(\frac{t}{w}\right)^{c_1} \left(\frac{page}{t}\right)^{c_2} mem^{c_3} * f^{c_4} + \left(\frac{t}{w}\right)^{c_5} \left(\frac{n}{t}\right)^{c_6} mem^{c_7} * f^{c_8} \right] \quad (9)$$

When the amount of data n , the number of pages $page$, the size of running memory mem and the number of iterations k of the algorithm are known, the model can have an optimal value by adjusting the number of tasks t , the number of cluster nodes w and the frequency of CPU f . For example, when $page = 5000000$, $n = 20000000$, mem is 20 GB, algorithm iterations k is 8, and the frequency range is [1800, 2600] MHz, the

minimum value of the energy consumption model which calculated by 1stOpt software [12] is 415483.6 J. At this time, the number of tasks is 677, the number of nodes is 2, and the frequency is 2560 MHz. In our known experimental data, the optimal energy consumption of the same data scale is 392197.61 J, and the number of tasks is 700, the number of nodes is 2, and the frequency is 2600 MHz. The optimal value predicted by the model is very close to the optimal value of actual measurement.

Therefore, in the case of known data scale and operating environment, we can learn how to adjust the number of tasks t , node number w and CPU frequency f to achieve the theoretical optimal value of energy consumption. Although there are some errors compared with the actual measured value, the optimal energy consumption value can be obtained faster and better for the case of large amounts of data.

5 Conclusion

The characteristics of big data applications are complex, and there are many factors affecting energy efficiency. We combine the system characteristics with the application characteristics to build the energy consumption model for the PageRank application. Based on the PageRank workload of big data basic test suite HiBench, we collect energy monitoring data for applications with different dataset scales and build models for the energy consumption. According to this model, we can locate the optimal configuration more quickly and make the energy consumption reach its optimal value. The experimental results show that the energy consumption model can obtain reasonable accuracy and the optimization method is effective. Future work may include trying to use other machine learning algorithms to improve the prediction accuracy of the model, as well as studying an energy consumption model for heterogeneous clusters.

Acknowledgements. The work was sponsored by the key research and development plan of Shaanxi Province. NO. 2019ZDLGY17-02 and the Fundamental Research Funds for the Central Universities.

References

1. Kavanagh, R., Djemame, K.: Rapid and accurate energy models through calibration with IPMI and RAPL. *Concurrency Comput. Pract. Exp.* **31**(13), 1–21 (2019)
2. Apache Spark Homepage. <https://spark.apache.org/>.
3. Lim, S.H., Sharma, B., Tak, B.C., Das, C.R.: A dynamic energy management scheme for multi-tier data centers. In: *IEEE International Symposium on Performance Analysis of Systems and Software*, Austin, TX, pp. 257–266. IEEE (2011).
4. Masdari, M., Nabavi, S.S., Ahmadi, V.: An overview of virtual machine placement schemes in cloud computing. *J. Network Comput. Appl.* **66**, 106–127 (2016)
5. Mann, Z.A.: Allocation of virtual machines in cloud data centers—a survey of problem models and optimization algorithms. *ACM Comput. Surv.* **48**(1), 1–34 (2015)
6. Luo, L., Wu, W.J., Zhang, F.: Energy modeling based on cloud data center. *J. Softw.* **25**(7), 1371–1387 (2014)

7. Dayarathna, M., Wen, Y., Fan, R.: Data center energy consumption modeling: a survey. *IEEE Commun. Surv. Tutor.* **18**(1), 732–794 (2017)
8. Wu, W.T., Lin, W.W., Hsu, C.H., He, L.G.: Energy-efficient hadoop for big data analytics and computing: a systematic review and research insights. *Future Gener. Comput. Syst. Int. J. eScience* **86**, 1351–1367 (2018)
9. Mashayekhy, L., Nejad, M.M., Grosu, D., Zhang, Q., Shi, W.S.: Energy-aware scheduling of mapreduce jobs for big data applications. *IEEE Trans. Parallel Distrib. Syst.* **26**(10), 2720–2733 (2015)
10. Li, H., Wang, H., Fang, S., Zou, Y., Tian, W.: An energy-aware scheduling algorithm for big data applications in Spark. *Cluster Comput.* **23**(2), 593–609 (2019). <https://doi.org/10.1007/s10586-019-02947-9>
11. Wang, Y.L., Zhao, T.H., Li, L., Hou, Z.X., Gu, J.H.: Roofline model based performance-aware energy management for scientific computing. In: 9th International Conference on Parallel Architectures, Algorithms and Programming, Taiwan, pp. 74–80. IEEE (2018)
12. 1stOpt Homepage. <https://www.7d-soft.com/>



Distributed Dense Tucker Decomposition Based on Hierarchical SVD

Zisen Fang^{1,2}, Fumin Qi¹✉, Yichuan Dong¹, Yong Zhang²,
and Shengzhong Feng¹

¹ National Supercomputing Center in Shenzhen, Shenzhen, China
{fangzs, qifm, dongyc, fengsz}@nscsz.cn

² Shenzhen Institutes of Advanced Technology, Chinese Academy of Science,
Beijing, China
zhangyong@siat.ac.cn

Abstract. As an important tool of multi-way/tensor data analysis tool, Tucker decomposition has been applied widely in various fields. But traditional sequential Tucker algorithms have been outdated because tensor data is growing rapidly in term of size. To address this problem, in this paper, we focus on parallel Tucker decomposition of dense tensors on distributed-memory systems. The proposed method uses Hierarchical SVD to accelerate the SVD step in traditional sequential algorithms, which usually takes up most computation time. The data distribution strategy is designed to follow the implementation of Hierarchical SVD. We also find that compared with the state-of-the-art method, the proposed method has lower communication cost in large-scale parallel cases under the assumption of the α - β model.

Keywords: Dense tensor · Tucker decomposition · Distributed memory

1 Introduction

In computer science, tensors are N-way arrays. In early days, tensor data was dealt with by vectorization, which broke construction information of original data. With the development of higher order data processing, tensor analysis has drawn lots of attention and shown advantage to vectorization methods. As the most important tool in tensor analysis, tensor decomposition methods are widely applied in multi fields such as computer vision [1, 2], machine learning [3, 4], neuroscience [5], data mining [6], social network computing [7] and so on. There are several tensor decomposition methods, among which Tucker decomposition [8] is one of most famous ones.

Tucker decomposition is traditionally solved by HOOI [9] with quite expensive cost due to iteration progress and TTM (Tensor Times Matrix) chain computation. Since Higher Order Singular Values Decomposition (HOSVD) [10] plays the role of initial step in HOOI, its truncated version Truncated HOSVD (T-HOSVD) can be used to approximate the output of HOOI, with only a little accuracy lost but great reduction of computation cost. An even better approximation is Sequential Truncated HOSVD (ST-HOSVD) [11], which applies greedy strategy to the truncation progress and achieves improvements in both accuracy and speed.

Although ST-HOSVD has reduced lots of computation cost compared to traditional methods like HOOI, it's still far from enough. With the development of our society and techniques, data sizes are growing rapidly. Traditional sequential solution within one computation node fails to handle Tucker decomposition of large tensor data today, due to not only extremely long computation time but also memory limit of one single node. To solve Tucker decomposition efficiently, recent years have seen multiple parallel algorithms on this issue.

Based on Hadoop [12], HaTen2 [13] can handle sparse tensor data of billion scale. This method divides complex tensor multiplications into large amount of unrelated vector dot products. To cut the dependencies of original sequential steps, HaTen2 takes additional operators which increases computation cost but improves parallelism a lot. For sparse tensors, this method reduces execution time on clusters with few additional tasks, while it costs much more for nonsparse cases. To deal with dense data, [14] proposes the first distributed dense Tucker decomposition algorithm on CPU clusters. It distributes tensor data across all modes. For example, let \mathcal{X} be a tensor of size $I_1 \times I_2 \times I_3$, and $p = p_1 \times p_2 \times p_3$ be the number of available processors. \mathcal{X} is divided into tensor blocks of size $\frac{I_1}{p_1} \times \frac{I_2}{p_2} \times \frac{I_3}{p_3}$, and each processor owns one block. This distribution strategy has been followed by several works as described below. [15] distributes tensor data using the same strategy and optimizes the TTM chain computation for parallel HOOI. It constructs TTM-trees to present TTM schemes and searches the best one with least computational load. GPUtensor [16] applies the same distribution strategy of [14] to Tucker decomposition on GPU platforms.

This paper focus on Tucker decomposition of dense tensors, in which case, [14] offers the current state-of-the-art method. Although there are some other methods extends it, they all follow its data distribution strategy. But actually, it has not optimized communication cost among different processors yet, which has a great influence on computation efficiency for parallel algorithms. As the data size increases day by day, large scale parallel techniques are in great need for data analysis. So in this paper, we present a new parallel Tucker decomposition method with optimized communication cost. Different from the methods mentioned above, we merge the gram matrix computation and eigen decomposition into one SVD step, which is implemented in parallel based on Hierarchical SVD [17] and leads to much less communication cost.

The rest of this paper is organized as follows. Section 2 is the preliminaries including basic concepts of tensor operations and a simple instruction of ST-HOSVD. Section 3 explains the details of the proposed method, focusing on implementation of parallel SVD and communication cost analysis. Section 4 is the experiments.

2 Preliminaries

2.1 Tensor Notations

In this paper, we denote a tensor by calligraphic letters, like \mathcal{X} . The dimension of a tensor is called order (also called mode). The space of a real tensor of order N and size $I_1 \times \dots \times I_N$ is denoted as $R^{I_1 \times I_2 \times \dots \times I_N}$.

Definition 1 (Matricization). The mode- n matricization of tensor $\mathcal{X} \in R^{I_1 \times I_2 \times \dots \times I_N}$ is the matrix, denoted as $\mathcal{X}_{(n)} \in R^{I_n \times \prod_{k \neq n} I_k}$, whose columns are composed of all the vectors obtained from \mathcal{X} by fixing all indices but n -th.

Definition 2 (Folding Operator). Suppose \mathcal{X} be a tensor. The mode- n folding operator of a matrix $M = \mathcal{X}_{(n)}$, denoted as $fold_n(M)$, is the inverse operator of the unfolding operator.

Definition 3 (Mode- n Product, TTM operator, Tensor Times Matrix). Mode- n product of tensor $\mathcal{X} \in R^{I_1 \times I_2 \times \dots \times I_N}$ and matrix $A \in R^{J \times I_n}$ is denoted by $\mathcal{X} \times_n A$, whose size is $I_1 \times \dots \times I_{n-1} \times J \times I_{n+1} \times \dots \times I_N$, and defined by $\mathcal{X} \times_n A = fold_n(M)$.

Definition 4 (Gram Matrix). The Gram Matrix of a matrix M is MM^T .

2.2 St-HOSVD

Tucker factorization is a method that decomposes a given tensor $X \in R^{I_1 \times I_2 \times \dots \times I_N}$ into N factor matrices U_1, U_2, \dots, U_N and a core tensor \mathcal{C} such that

$$\mathcal{X} \approx \mathcal{C} \times_1 U_1 \dots \times_N U_N \quad (1)$$

In this paper, we consider the optimization of ST-HOSVD [10] to solve Tucker decomposition. The algorithm of ST-HOSVD is described as Algorithm 1 below.

Algorithm 1. ST-HOSVD[10]
Input: The tensor $\mathcal{X} \in R^{I_1 \times I_2 \times \dots \times I_N}$, ranks (r_1, r_2, \dots, r_N)
Output: The core tensor \mathcal{C} , and factor matrices U_1, U_2, \dots, U_N
$\mathcal{C} \leftarrow \mathcal{X}$
For $k = 1 \dots N$
Calculate Gram matrix : $G = \mathcal{C}_{(k)} \cdot \mathcal{C}_{(k)}^T$
Take the leading r_k eigen vectors of G and form the factor matrix U_k
$\mathcal{C} \leftarrow \mathcal{C} \times_k U_k$
EndFor

3 The Proposed Method

In this section, we will describe the proposed method in detail. In ST-HOSVD (Algorithm 1), to obtain the factor matrix of every mode, one needs to calculate the gram matrix of $\mathcal{X}_{(n)}$ and its eigen decomposition. But actually, one SVD step is sufficient to obtain the factor matrix. What's more, in parallel cases, using truncated SVD can reduce communication cost, as is explained later. This is also a reason of why we prefer SVD here.

The implementation of parallel SVD in the proposed method can be divided into 3 steps:

- Distribute data to each processor.
- Compute SVD of the local data on each processor.
- Root processor gathers SVD results and merge them to get the factor matrix.

Step 2 and 3 are the key of the proposed method, and data distribution in step 1 should follow them. So Sect. 3.1 explains step 2 and 3 firstly, followed by Sect. 3.2 with accuracy analysis. Section 3.3 describes data distribution and analysis communication cost after that.

3.1 Hierarchical SVD

The matricization of a tensor is usually a highly rectangle matrix $M \in R^{I \times D}$ with $I \ll D$. To compute the SVD of M efficiently, a natural way is to divide M into smaller matrices such that $M = [M_1|M_2|...|M_p]$ with $M_i \in R^{I \times D_i}$, and compute SVD for each M_i and then merge them. Actually, this is the exact idea of Hierarchical SVD [17], which computes U and S parallelly while discarding V for the SVD of $M : U \cdot S \cdot V^T$. And we can see in Algorithm 1, in ST-HOSVD, to obtain the factor matrix of Tucker decomposition for mode n , one only needs the left singular vectors U from the SVD of $\mathcal{X}_{(n)} = U \cdot S \cdot V^T$.

The Hierarchical SVD algorithm calculates the SVD of each M_i by $U_i S_i V_i^T$ and discards the right singular matrices V_i^T . Then it calculates the SVD of $[U_1 S_1 | U_2 S_2 | ... | U_p S_p]$ and output the left singular matrix and singular values. Figure 1 illustrates the progress of Hierarchical SVD.

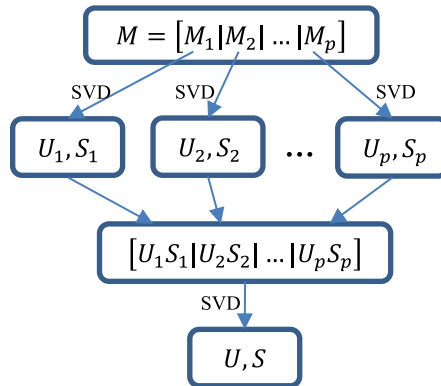


Fig. 1. The illustration of Hierarchical SVD [17].

In the real world, most tensor data are of low-rank, which means truncated SVD with only a few leading singular values may lose little information of the original data.

In this case, using SVD instead of gram matrix can reduce communication cost. The reason runs as follows. Suppose every processor has a local matrix $M_i \in R^{I \times D_i}$. The local gram matrix is of size $I \times I$, which is the transferring data size per processor. But if we use SVD and the truncated rank is d , then each processor only needs to transfer $U_i S_i \in R^{I \times d}$. As mentioned above, d is usually much smaller than I in real world data, so the transferring data size $I \times d$ is also much smaller than that of local gram matrix computation.

In extremely large-scale parallel cases with very large p , one can calculate SVD with more levels to reduce communication cost further. See Fig. 2 as an example. But one should also note that more levels come with larger error for truncated SVD. So we suggest this strategy only serve to trade off accuracy for computation time, or cases of non-truncated SVD.

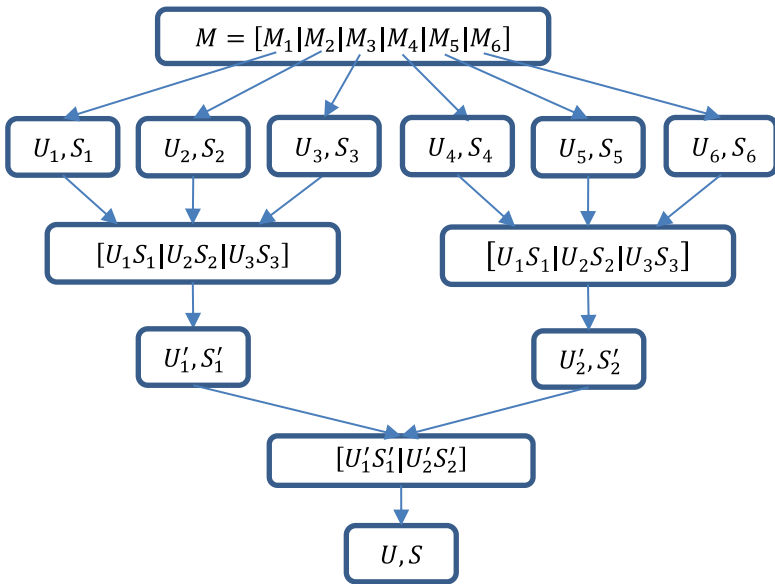


Fig. 2. Example of two-level Hierarchical SVD.

3.2 Discussion on Error of Hierarchical SVD

Now let's come to the accuracy of Hierarchical SVD. For normal SVD, the error analysis is very clear. Let A be a matrix and s_i, u_i, v_i ($i = 1, \dots, n$) be its singular values in descent order and the corresponding left singular vectors and right singular vectors. Let $(A)_d$ represents the recovered matrix of reduced SVD of A with rank $d \leq n$, that is,

$$(A)_d = \sum_{i=1}^d s_i \cdot u_i \cdot v_i^T \tag{2}$$

Then $(A)_d$ is the best approximation of A such that the rank is at most d . And the approximation error is $\|A - (A)_d\|_F = \sqrt{\sum_{i=d+1}^n s_i^2}$.

In [17], there is also a theorem to guarantee the error bound of Hierarchical SVD with multiple layers. We present it here as below.

Theorem 1 (from [17]). Let A be a matrix. Assume that U and S are the outputs of q -level Hierarchical SVD algorithm with input A . Then there exists a unitary matrix V such that $US = AV + \Psi$, where

$$\|\Psi\|_F \leq \left[(1 + \sqrt{2})^{q+1} - 1 \right] \|A - (A)_d\|_F \tag{3}$$

From Theorem 1, $USV^T = A + \Psi V^T$, and $\|\Psi V^T\|_F = \|\Psi\|_F$, then we can know that U from Hierarchical SVD is the normal left matrix of $A + \Psi V^T$, which is close to A . This means that Hierarchical SVD may increase error of reduced SVD by at most $\left[(1 + \sqrt{2})^{q+1} - 1 \right]$ times.

For 1-layer Hierarchical SVD, there is a tighter bound guaranteed by another theorem from [17] as below. Here, we denote $\bar{A} := US$, where the SVD of A is $A = USV^T$.

Theorem 2. Let $A = [A_1|A_2|\dots|A_p]$, $B = \left[\overline{(A_1)_d} | \overline{(A_2)_d} | \dots | \overline{(A_p)_d} \right]$, and $B' = B + \Psi$, then there exists a unitary matrix W such that

$$\left\| A - \overline{(B')_d} W \right\|_F \leq 3\sqrt{2} \|A - (A)_d\|_F + (1 + \sqrt{2}) \|\Psi\|_F \tag{4}$$

In Theorem 2, it is routine to see that $\overline{(B)_d} = US$, where U and S are the outputs of 1-level Hierarchical SVD algorithm with input A . Setting Ψ to zero matrix will lead to

$$\left\| A - \overline{(B)_d} W \right\|_F \leq 3\sqrt{2} \|A - (A)_d\|_F \tag{5}$$

So for 1-layer Hierarchical SVD, Hierarchical SVD may increase error of reduced SVD by at most $3\sqrt{2}$ times, which is small than the bound in Theorem 1 $\left[(1 + \sqrt{2})^2 - 1 \right] = 2 + 2\sqrt{2}$.

3.3 Data Distribution and Communication Cost

Consider a tensor $\mathcal{X} \in R^{I_1 \times I_2 \times \dots \times I_N}$. Denote $I = I_1 \times I_2 \times \dots \times I_N$ be the number of entries of the tensor, and $J_k = \frac{I}{I_k}$ be the products of all I_i except for I_k .

In STHOSVD, data distribution occurs when implementing Hierarchical SVD across processors. For every mode k , the proposed method applies Hierarchical SVD on $\mathcal{X}_{(k)}$, the matricization of the tensor. We distribute data in a natural way. Divide $\mathcal{X}_{(k)} = [M_1|M_2|\dots|M_p]$, where $M_i \in R^{I_k \times D_i}$ and $\sum_{i=1}^p D_i = J_k$, and every processor

owns one matrix block M_i . To achieve maximum performance, the matrix blocks should have sizes of almost the same.

Assume that M_1, M_2, \dots, M_p happen to have the same size of $I_k \times \frac{I_k}{p}$. Using the α - β model (the latency cost vs the per-word transfer cost), the communication cost of distributing M_1, M_2, \dots, M_p from root processor to other processors is $(p-1)\alpha + I_k \frac{I_k}{p} \beta = (p-1)\alpha + \frac{I}{p} \beta$. After every processor finishes calculating the SVD of the assigned matrix block M_i , the root processor will gather $U_i S_i$ from all processors, and the communication cost is $\alpha \log_2(p) + (p-1)I_k r_k \beta$, where r_k is the reduced rank of mode k . So the total communication cost when calculating factor matrix of mode k is $(p-1)\alpha + \frac{I}{p} \beta + \alpha \log_2(p) + \frac{(p-1)}{p} I_k r_k \beta = (p-1 + \log_2(p))\alpha + \left[\frac{I}{p} + \frac{(p-1)}{p} I_k r_k \right] \beta$.

Consider the state-of-the-art method from [14], with p processors organized in a grid of size $p_1 \times p_2 \times \dots \times p_N$. Calculation of factor matrix of mode k includes an MPI send-receive operator and reduce operator for gram matrix, see [14] for details. The communication cost is $(p_k - 1) \left(\alpha + \frac{I}{p} \beta \right) + \alpha \log_2(p_k) + \frac{(p_k - 1)}{p} I_k^2 \beta = (p_k - 1 + \log_2(p_k))\alpha + \left[(p_k - 1) \frac{I}{p} + \frac{(p_k - 1)}{p} I_k r_k \right] \beta$. For most cases, $\frac{I}{p}$ is the dominant item. In large-scale parallel cases, p is very large so that every p_k is large too. Comparing the proposed method with the state-of-the-art method from [14], it is easy to see that the former one has lower communication cost.

TTM operator is simple here. We need only to send the factor matrix U_k from root to all processors and the latter ones update the local assigned matrix block by $M_i \leftarrow M_i \times_k U_k$. Here we ignore the communication cost cause it needs little data to transfer, and the case of the state-of-the-art method from [14] is the same.

The proposed distributed ST-HOSVD algorithm is summarized in Algorithm 2.

Algorithm 2. The proposed Distributed ST-HOSVD

Input: The tensor $\mathcal{X} \in R^{I_1 \times I_2 \times \dots \times I_N}$, ranks (r_1, r_2, \dots, r_N)

Output: The core tensor \mathcal{C} , and factor matrices U_1, U_2, \dots, U_N

FOR $k = N, N-1, \dots, 1$

Distribute data to all processors

Hierarchical SVD step:

Every processor calculates the reduced SVD (rank r_k) of the assigned matrix block by $M_i \leftarrow U_i S_i V_i^T$

Root processor gathers $U_i S_i$ from all processors

Root processor calculates the reduced SVD (rank r_k) of $[U_1 S_1 | U_2 S_2 | \dots | U_p S_p] \leftarrow USV^T$

Update factor matrix $U_k \leftarrow U$

Broadcast U_k to each processor from root.

Every processor updates the assigned matrix block by $M_i \leftarrow M_i \times_k U_k$

ENDFOR

Gather all M_i to root processor and fold them to form the tensor core tensor \mathcal{C} .

4 Experiments

We conduct two kinds of experiments in this section. The first one focuses on the merging results of Hierarchical SVD, and the second one focuses on the comparison of the proposed method and the state-of-the-art method.

4.1 Experiments of SVD

In this part, we generate a random matrix A of size 128×1024 . The matrix A is divided into different number of blocks by columns ($A = [A_1|A_2|\dots|A_p]$), and the 1-layer Hierarchical SVD with reduced rank 64 is implemented on it. Reduced SVD is used here as a comparison. In Hierarchical SVD, the right singular matrix V is discarded, and the left singular matrix U is the key output. To evaluate the quality of U , we estimate the projection error of A by

$$e = \frac{\|A - UU'A\|_F}{\|A\|_F} \quad (6)$$

For the normal reduced SVD, the error is 0.2962. For Hierarchical SVD, the results runs is presented in Table 1. We can see that the projection errors of Hierarchical SVD are very close to that of the normal reduced SVD. These results show Hiersarchical SVD has only little accuracy lost compared with normal reduced SVD.

Table 1. Projection Errors with different p .

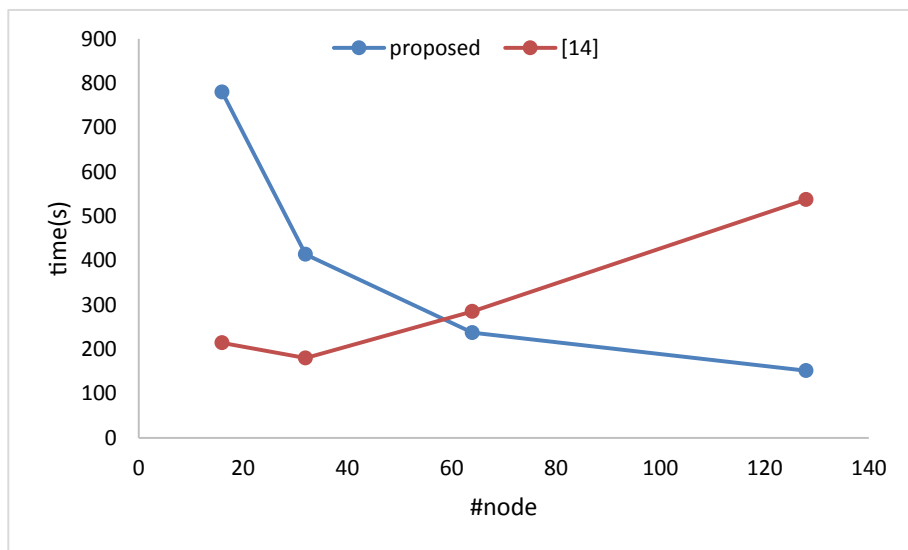
p	2	4	8	16
e	0.2986	0.2980	0.2969	0.2962

4.2 Experiments of Tucker Decomposition

In this part, we present some experimental results to verify the performance of the proposed method. The experiments are implemented on a cluster with 128 nodes. Each node is equipped with 128 GB memory, a 32-core CPU of 2.0 GHz and a GPU of 13.3 TFLOPs peak single precision (FP32) Performance. We generate five tensor of size $1024 \times 1024 \times 1024$ randomly, and compare the performances of the proposed method with the state-of-the-art method from [14]. The two methods are implemented using different number of nodes, namely 16, 32, 64, 128, to test their scalabilities. For each method, we apply it to the five tensors and set the reduced rank to be 512 for all modes. For the proposed method, the number of layers is set to 1 in Hierarchical SVD. The average executing time of each method is taken as its performance. The experimental result is reported in Table 2 and Fig. 3. It is clear that the proposed method has better performance with larger scale parallelism.

Table 2. The average executing time versus clusters of different number of nodes.

#nodes	Time (s)	
	Proposed	[14]
16	779.86	214.33
32	413.84	180.18
64	237.22	285.16
128	151.58	537.62

**Fig. 3.** The average executing time versus clusters of different number of nodes.

5 Conclusion

In this paper, we focus on the problem of high-performance Tucker decomposition on distributed-memory systems. We have proposed a method based on Hierarchical SVD for the problem, which has lower communication cost in large-scale cases compared with the state-of-the-art method. The experiments highlight the proposed method in term of executing time although there is a little accuracy lost.

In the future, we are going to explore how to make full use of the idea of Hierarchical SVD to improve parallel efficiency and reduce the error bound.

Acknowledgements. The work was supported by the National Key Research and Development Project of China (Grant No. 2019YFB2102500).

References

1. Barmpoutis, A.: Tensor body: real-time reconstruction of the human body and avatar synthesis from RGB-D. *IEEE Trans. Cybern.* **43**(5), 1347–1356 (2013)
2. Tao, D., Li, X., Wu, X., Maybank, S.J.: General tensor discriminant analysis and Gabor features for gait recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(10), 1700–1715 (2007)
3. Li, X., Lin, S., Yan, S., Xu, D.: Discriminant locally linear embedding with high-order tensor data. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **38**(2), 342–352 (2008)
4. Signoretto, M., Tran Dinh, Q., De Lathauwer, L., Suykens, J.A.K.: Learning with tensors: a framework based on convex optimization and spectral regularization. *Mach. Learn.* **94**(3), 303–351 (2013). <https://doi.org/10.1007/s10994-013-5366-3>
5. Mørup, M., Hansen, L.K., Herrmann, C.S., Parnas, J., Arnfred, S.M.: Parallel factor analysis as an exploratory tool for wavelet transformed event-related EEG. *NeuroImage* **29**(3), 938–947 (2006)
6. Mørup, M.: Applications of tensor (multiway array) factorizations and decompositions in data mining. *Wiley Interdisc. Rev. Data Mining Knowl. Disc.* **1**(1), 24–40 (2011)
7. Sun, J., Papadimitriou, S., Lin, C.-Y., Cao, N., Liu, S., Qian, W.: Multivis: content-based social network exploration through multi-way visual analysis. In: Proceedings of the 2009 SIAM International Conference on Data Mining, pp. 1064–1075. SIAM (2009)
8. Tucker, L.R.: Some mathematical notes on three-mode factor analysis. *Psychometrika* **31**(3), 279–311 (1966)
9. De Lathauwer, L., De Moor, B., Vandewalle, J.: On the best rank-1 and rank-(R1, R2, ..., RN) approximation of high-order tensors. *SIAM J. Matrix Anal. Appl.* **21**(4), 1324–1342 (2000)
10. Lathauwer, L.D., Moor, B.D., Vandewalle, J.: A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.* **21**, 1253–1278 (2000)
11. Vannieuwenhoven, N., Vandebril, R., Meerbergen, K.: A new truncation strategy for the higher-order singular value decomposition. *SIAM J. Sci. Comput.* **34**(2), A1027–A1052 (2012)
12. Hadoop information. <https://hadoop.apache.org/>
13. Jeon, I., Papalexakis, E.E., Kang, U., Faloutsos, C.: HaTen2: billion-scale tensor decompositions. In: 2015 IEEE 31st International Conference on Data Engineering, Seoul, 2015, pp. 1047–1058. <https://doi.org/10.1109/ICDE.2015.7113355>
14. Austin, W., Ballard, G., Kolda, T.G.: Parallel tensor compression for large-scale scientific data. In: 2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Chicago, IL, pp. 912–922 (2016). <https://doi.org/10.1109/IPDPS.2016.67>
15. Chakaravarthy, V.T., Choi, J.W., Joseph, D.J., Murali, P., Sreedhar, D.: On optimizing distributed tucker decomposition for sparse tensors. In: ICS 2018: Proceedings of the 2018 International Conference on Supercomputing, pp. 374–384 (2018)
16. Zou, B., Li, C., Tan, L., Chen, H.: GPUtensor: efficient tensor factorization for context-aware recommendations. *Inf. Sci.* **299**(4), 159–177 (2015)
17. Iwen, M.A., Ong, B.W.: A distributed and incremental SVD algorithm for agglomerative data analysis on large networks. *SIAM J. Matrix Anal. Appl.* **37**(4), 1699–1718 (2016)



A Multi-objective Task Offloading Strategy for Workflow Applications in Mobile Edge-Cloud Computing

Yongqiang Gao^(✉) and Dandan Yan

Inner Mongolia University, Hohhot, China
{csgyq, csydd}@outlook.com

Abstract. Prompted by the remarkable progress in both edge computing and cloud computing, mobile edge-cloud computing has become a promising computing paradigm, where mobile users may take advantages of the low-latency property of edge computing and the rich-resource capacity of cloud computing to provide a high quality of service for mobile applications. In mobile edge-cloud computing, a main challenge is how to efficiently offload workflow tasks in order to reduce energy consumption, and improve performance and reliability. In this paper, we formulate workflow offloading into a constrained multi-objective optimization problem and develop a hybrid algorithm involving differential evolution algorithm, artificial bee colony optimization and decoding heuristic to explore the optimal strategy of task offloading for workflow applications in mobile edge-cloud computing. The effectiveness of our strategy is evaluated by extensive simulations using real-world workflows. The results show that our strategy performs better than the state-of-the-art methods applied to similar problems.

Keywords: Multi-objective optimization · Mobile edge computing · Offloading strategy · Energy consumption · Reliability · Workflow

1 Introduction

With the large-scale popularization of the Internet of things, the demand for mobile devices is gradually increasing in all professions and trades, such as retail, health care, mobile payment, positioning navigation, natural language processing, which puts forward higher requirements for mobile terminals [1]. The execution of workflow application usually needs to consume a lots of physical resources and energy, but the physical resources, and battery capacity of mobile devices are limited. Therefore, it is difficult to meet the service quality requirements of workflow application when running the compute-intensive workflow tasks in mobile devices.

Task offloading technology is proposed for alleviating the pressure of physical resource on mobile devices and providing mobile users with a high quality of service [2]. In mobile cloud environment, some computing tasks in mobile applications can be offloaded from mobile terminals to remote cloud data center through wireless network or cellular network. However, remote cloud data centers are often far away from mobile users, which can cause a large transmission delay. In mobile edge computing

environment, some computing tasks in mobile applications can be offloaded to MEC servers deployed on the base station near user terminals, which can decrease waiting delay and improve service quality. But MEC servers in mobile edge environment can only provide relatively limited resources comparing with the high performance servers in mobile cloud environment [3]. Therefore, it is a major challenge for researchers to understand how to effectively offload workflow tasks by combining the advantages of cloud computing and edge computing so as to simultaneously optimize energy consumption, completion time and reliability.

A workflow usually consists of interdependent tasks. Traditional offloading methods focus on independent tasks, and thus it cannot apply to workflow offloading. In addition, the existing task offloading methods for mobile edge-cloud environment mainly focus on a single objective target, such as energy consumption, completion time, or reliability. To address these problems, we propose a multi-objective task offloading strategy based on a hybrid of differential evolution algorithm, artificial bee colony optimization and decoding heuristic for scheduling workflow tasks in mobile edge-cloud environment with the goal of optimizing the completion time, energy consumption and reliability. The contributions of this paper are summarized as follows:

- We formulated the task offloading for workflow in mobile edge-cloud environment as a constrained multi-objective optimization problem. Our proposed optimization models consider the energy consumption of mobile devices, the completion time of workflow and the reliability of task execution.
- We proposed a multi-objective hybrid algorithm, called HABC, for finding the best offloading strategy. A differential evolution operator is embedded to artificial bee colony algorithm in order to improve the diversity of food sources. In addition, a decoding heuristic is integrated into artificial bee colony algorithm to generate a feasible task offloading strategy of workflow application.
- We evaluate the effectiveness of the proposed offloading strategy by various simulation experiments. The results show, compared with other alternatives, our method can achieve lower energy consumption, shorter completion time and higher reliability.

The remaining section of the paper is arranged as follows. A survey on related work is presented in Sect. 2. Section 3 introduces the model and the expression of the problem. The hybrid algorithm in detail is introduced by Sect. 4. Section 5 gives the experimental results. Finally, the conclusion is drawn in Sect. 6.

2 Related Work

This paper focus on task offloading of workflow in mobile edge-cloud computation. We review the related work from the following two aspects: some research in mobile cloud computing (MCC) and some achievement in mobile edge computing.

Recently, many efforts have been devoted to the computation offloading problem in MCC. Wu et al. [4] proposed a scheduling scheme of task in MCC, which applied dynamic voltage frequency scaling technology and whale optimization algorithm, and considered three factors including task execution location, task execution sequence and

mobile device operating voltage frequency to optimize completion time and energy consumption. Hu et al. [5] proposed a task scheduling method, which adopted simulate anneal algorithm to save the energy consumption of mobile devices and improve the completion time of tasks. Pu et al. [6] proposed a task scheduling scheme based on genetic algorithm to allocate computing resources reasonably so as to optimize energy consumption, reliability, overall cost and user experience.

There are also some efforts focusing on offloading computation tasks to mobile edge computing platform. Fu et al. [7] proposed a task computing migration method aiming at a large amount of energy consumption of mobile devices in mobile edge environment, and established time model and energy consumption model, then allocated reasonably the computing tasks of workflow by non-dominated sorting algorithm. Mao et al. [8] investigated a green MEC system with energy harvesting devices and develop an effective computation offloading strategy. The execution cost, which addresses both the execution latency and task failure, is adopted as the performance metric. A low-complexity online algorithm is proposed. Kong et al. [9] proposed the fine-grained migration strategy, which can effectively reduce the energy consumption of mobile terminal devices on the premise of meeting the task execution delay by genetic algorithm, giving full play to the advantages of mobile edge computing.

Different from the above studies, we propose a hybrid algorithm based on differential evolution algorithm, artificial bee colony optimization and decoding heuristic for workflow task offloading in mobile edge-cloud computing with the goal of optimizing energy consumption of mobile terminal, completion time of workflow and reliability of execution task.

3 Problem Model

3.1 System Architecture

As shown in Fig. 1, mobile edge-cloud computing system consists of three layers: mobile terminal, edge server and cloud data center. Workflow tasks can be offloaded to the server deployed around the base station and the wireless access point. When the resources of edge server is not enough to satisfy the user needs, workflow tasks are offloaded to cloud data center. Mobile edge-cloud computing combines cloud computing and edge computing to expand the computing capacity of mobile terminal and improve the quality of user experience.

3.2 Workflow Model

Generally, workflow application is defined as a directed acyclic graph (DAG), and described by a two-tuple $G = (T, R)$, where $T = \{t_1, t_2, \dots, t_n\}$ represent a set of n different tasks in a workflow and R represents the set of dependency between workflow tasks. Each dependency has a weight $d_{i,j}$ that represents the amount of data transmission between two tasks. For a given dependency $r_{i,j}$ between task t_i and task t_j , task t_i is known as one of the predecessors of task t_j , and task t_j is known as one of the successors of task t_i . Every task may has several predecessors and successors tasks

besides entry task and exit task. Task t_j cannot start its execution before the results of its all predecessors are transferred to it.

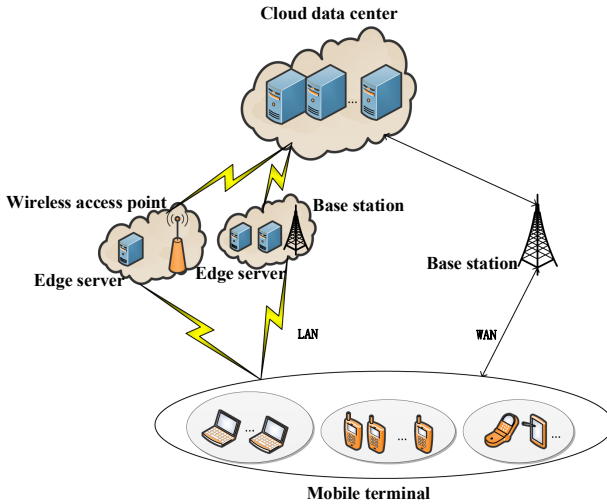


Fig. 1. Architecture diagram of mobile edge-cloud computing.

3.3 Computation Time Model

Completion time of workflow depend on task waiting time, task execution time and data transmission time. We apply M/M/N queuing system [10] to get the average waiting time W_q .

In the mobile edge-cloud system, the workflow task can be performed in mobile terminal, edge server and cloud server. Therefore, the execution time D_{exe} of task t_i can be defined as:

$$D_{exe}(t_i) = \begin{cases} \frac{wl_i}{f_m} & m_i = 0 \\ \frac{wl_i}{f_m} + W_q^e + D_{lan} & m_i = 1 \\ \frac{wl_i}{f_c} + W_q^c + D_{wan} & m_i = 2 \end{cases} \quad (1)$$

where $m_i = 0$ indicates that the task t_i is performed on the mobile terminal, and $m_i = 1, 2$ represent that t_i is offloaded to the edge server and the cloud respectively. wl_i represents the workload of task t_i . f_m represents the CPU capacity of mobile terminal and f_e denotes the processing capacity of edge server. f_c represents the processing capacity of remote cloud. D_{lan} indicates the transmission delay in local area network (LAN) and D_{wan} indicated the transmission delay in wide area network (WAN).

When the task t_i and the task t_j are performed in the same place or two tasks transmit data between edge server and the cloud, the transmission time of task is defined as:

$$D_{tran}(t_i, t_j) = 0 \quad (2)$$

If two tasks are performed on the mobile terminal and the edge server respectively, the data is transmitted through LAN. The transmission time of task is calculated as:

$$D_{tran}(t_i, t_j) = \frac{d_{ij}}{B_e} \quad (3)$$

where B_e indicates the bandwidth of LAN. If two tasks are performed on the mobile terminal and the cloud, the data is transmitted through WAN. The transmission time of task is calculated as:

$$D_{tran}(t_i, t_j) = \frac{d_{ij}}{B_c} \quad (4)$$

where B_c indicates the bandwidth of WAN.

The start time $D_{st}(t_i)$ and completion time $D_f(t_i)$ of task t_i are computed recursively as shown in (5) and (6).

$$D_{st}(t_i) = \begin{cases} D_{ent} & pred(t_i) = \emptyset \\ \max \left\{ \max_{t_k \in pred(t_i)} (D_{tran}(t_k, t_i) + D_f(t_k)), D_{ent} \right\} & otherwise \end{cases} \quad (5)$$

$$D_f(t_i) = D_{st}(t_i) + D_{exe}(t_i) \quad (6)$$

In (5), D_{ent} represents earliest time of task execution, and $pred(t_i)$ denotes all predecessors of task t_i . Finally, the completion time $D(M)$ of workflow is denoted as:

$$D(M) = \max_{i \in \{1, 2, \dots, n\}} D_f(t_i) \quad (7)$$

where M represents offloading decision of workflow.

3.4 Energy Consumption Model

The total energy consumption for mobile terminal contains the execution energy consumption and transmission energy consumption. Let $E_{exe}(t_i)$ be energy consumption generated by processing task t_i and $E_{tran}(t_i, t_j)$ be energy consumption generated by transmitting data from task t_i to task t_j . $E_{exe}(t_i)$ is given as:

$$E_{exe}(t_i) = \begin{cases} \frac{wl_i}{f_m} \times P_{act} & m_i = 0 \\ \left(\frac{wl_i}{f_e} + W_q^e + D_{lan} \right) \times P_{id} & m_i = 1 \\ \left(\frac{wl_i}{f_c} + W_q^c + D_{wan} \right) \times P_{id} & m_i = 2 \end{cases} \quad (8)$$

where P_{act} denotes the power when mobile terminal is active and P_{id} indicates the power when mobile terminal is idle.

If two dependent tasks transmit data by LAN, $E_{tran}(t_i, t_j)$ is calculated as:

$$E_{tran}(t_i, t_j) = \frac{d_{i,j}}{Be} \cdot P_{tran} \quad (9)$$

If two dependent tasks transmit data by WAN, $E_{tran}(t_i, t_j)$ is calculated as:

$$E_{tran}(t_i, t_j) = \frac{d_{i,j}}{Bc} \cdot P_{tran} \quad (10)$$

where P_{tran} represents the transmission power of mobile terminal. Therefore, the total energy consumption $E(M)$ of mobile terminal is defined by:

$$E(M) = \sum_{t_i \in T} E_{exe}(t_i) + \sum_{r(t_i, t_j) \in R} E_{tran}(t_i, t_j) \quad (11)$$

3.5 Reliability Model

The reliability of task is the probability of performing task successfully. During task execution, a fault is inevitable due to hardware failure, software bugs and so on. Therefore, we take into account the reliability of task execution. Inspired by the reliability model in [11], the reliability $R(t_i)$ of task t_i can be defined as:

$$R(t_i) = \begin{cases} e^{-\Lambda(f_m) \times \frac{c_m(t_i)}{f_m}} & m_i = 0 \\ e^{-\Lambda(f_e) \times \frac{c_e(t_i)}{f_e}} & m_i = 1 \\ e^{-\Lambda(f_c) \times \frac{c_c(t_i)}{f_c}} & m_i = 2 \end{cases} \quad (12)$$

$$\Lambda(f) = \Lambda_0 \cdot g(f) = \Lambda_0 \cdot 10^{\frac{d(1-f)}{1-f_m}} \quad (13)$$

where d and Λ_0 are constant. $c_m(t_i)$, $c_e(t_i)$ and $c_c(t_i)$ are the computation cost of mobile terminal, edge server and cloud sever respectively. The computation cost of cloud servers is the highest. Finally, a workflow unreliability $POF(M)$ can be defined as:

$$POF(M) = 1 - R(M) = 1 - \prod_{i=1}^n R(t_i) \quad (14)$$

where $R(M)$ is defined as a workflow reliability.

3.6 Problem Definition

The goal of workflow tasks offloading is to optimize energy consumption of mobile terminal, completion time of workflow and reliability of execution task. This workflow tasks offloading problem is formalized as:

$$\text{Minimize } E(M) \tag{15}$$

$$\text{Minimize } D(M) \tag{16}$$

$$\text{Minimize } POF(M) \tag{17}$$

Subject to

$$D_f(t_k) < D_{st}(t_i) \quad \forall t_i \in T, \forall t_k \in pred(t_i) \tag{18}$$

$$D_f(t_i) < D_{st}(t_j) \quad \forall t_i \in T, \forall t_j \in succ(t_i) \tag{19}$$

$$D_f(t_k) + D_{tran}(t_k, t_i) \leq D_{st}(t_i) \forall t_i \in T, \forall t_k \in pred(t_i) \tag{20}$$

$$m_i \in \{0, 1, 2\} \quad \forall t_i \in T, m_i \in M \tag{21}$$

Constraint (18)–(20) ensures that each task can only be performed after its predecessor has completed and all the needed data is received. $succ(t_i)$ represents all successors of task t_i . Constraint (21) defines the variables domain of the problem, where m_i is the decision variables.

4 The Proposed Hybrid Algorithm

The optimization problem constructed above belongs to the NP-Hard problem, so there is a great difficulty to find optimal solution in a short time. We develop a hybrid algorithm based artificial bee colony (ABC) and differential evolution algorithm to find good solutions in large solution space. The process of the HABC algorithm is described in Algorithm 1.

In the hybrid algorithm, a solution is represented by an array, and each index of the array represent a workflow task. The value of the array elements, which is a real number between 0 and 3, represents the offloading destination of the corresponding task. The array length is the number of workflow task. The decoding process generates a feasible workflow offloading strategy. We convert the solution into a feasible workflow offloading strategy by Algorithm 2.

The initial population is generated by a random strategy. After the initialization phase, the employed bees search for new food sources in the around of current food source. In this paper, each employed bee searches for a new solution v_i according to the elite oriented search strategy proposed in [12]. If the new solution v_i dominates the original food source x_i , the new solution v_i replaces the original food source x_i , and the trial for this new solution is set to zero. Otherwise, the trial for the original solution is

incremented by one. After all employed bees complete their search, they share information of food source with onlooker bees. We apply roulette rules to choose a food source, which is called target vector, and then update it through mutation operator, crossover operator and selection operator in differential evolution.

First, three different individuals different from target vector are chosen among the population, and then they are performed the mutation operation to generate a donor vector v_i^g . Next, the crossover operation is performed to improve the population variety. The target vector x_i^g and the donor vector v_i^g is crossed through binomial cross operation to generate a trial vector u_{ij}^g . Finally, the selection operation is performed to get the better vector.

A scout bee can increase the population diversity and enhance the capacity of jump out of local optimum. If a food source still cannot be improved through the number of predefined trial, the food source will be discarded. The employed bees becomes a scout bee after discarding food source, and the food source is replaced with a new solution which is produced in the same way as that in the initialization stage.

Algorithm 1. The proposed HABC algorithm

```

Input: Workflow, set of execution destination  $m_i$ 
Output: Set of Pareto solutions
1. Randomly generate an initial population with  $FN$  food sources
   Repeat
/*employed bee phase*/
2. For each bee in the current population generate a new food source
3. If the new food source dominates the original food source
4. replace the original food source with the new food source
5. the trial for the new food source is set to zero
6. Else
7. the trial for the original food source is increased by one
8. EndIf
9. EndFor
/* onlooker bee phase*/
11. For  $i=1$  to  $FN$ 
12. select a target vector according to the roulette rule
13. apply the mutation operation to the target vector
14. apply the crossover operation to the target vector
15. apply the selection operation to the target vector
16. EndFor
17. generate a new population by choosing  $FN$  food sources from the current population
/*scout bee phase*/
18. For each bee in the current population
19. If the trial exceeds the predefined value  $limit$ 
20. generate a new food source
21. replace the old food source with the new food source
22. the trial for the new food source is set to zero
23. EndIf
24. EndFor
25. Until the maximum number of iterations is reached
26. Return all non-dominated solutions in the current population
    
```

Algorithm 2. The decoding heuristic

```

Input: An encoding array, the set of edge servers, the set of cloud servers
Output:  $M=\{S,E,D,R\}$ 
1.  $S = \phi$   $D = \emptyset$   $E = \emptyset$   $R = \emptyset$ 
2. For  $i=1$  to  $n$ 
3. If  $t_i$  has no predecessor tasks
4.  $D_{st}(t_i) = D_{ent}$ 
5. Else
6.  $D_{st}(t_i) = \max \left\{ \max_{(t_k \in \text{pred}(t_i))} (D_{tran}(t_k, t_i) + D_j(t_k)), D_{ent} \right\}$ 
7. End If
8. If  $t_i$  performed on mobile terminal  $MT$ 
9.  $D_{exe}(t_i) = \frac{w_{i1}}{f_m}$ 
10.  $R(t_i) = e^{-\Delta(f_m) \times \frac{c_m(t_i)}{f_m}}$ 
11. Add the mapping of  $t_i$  to  $MT$  to  $S$ 
12. Else if  $t_i$  performed on edge server
13. allocate  $t_i$  to edge server  $EL_{t_i}$  with the least load
14.  $D_{exe}(t_i) = \frac{w_{i1}}{f_e} + W_q^e + D_{lan}$ 
15.  $R(t_i) = e^{-\Delta(f_e) \times \frac{c_e(t_i)}{f_e}}$ 
16. Add the mapping of  $t_i$  to  $EL_{t_i}$  to  $S$ 
17. Else if  $t_i$  performed on cloud
18. allocate  $t_i$  to cloud server  $CL_{t_i}$  with the least load
19.  $D_{exe}(t_i) = \frac{w_{i1}}{f_c} + W_q^c + D_{wan}$ 
20.  $R(t_i) = e^{-\Delta(f_c) \times \frac{c_c(t_i)}{f_c}}$ 
21. Add the mapping of  $t_i$  to  $CL_{t_i}$  to  $S$ 
22. End If
23. Calculating  $D_j(t_i)$  according to (6)
24. For each success task  $t_j$  of  $t_i$ 
25. If  $t_i, t_j$  are transmitted by  $LAN$ 
26.  $D_{tran}(t_i, t_j) = \frac{d_{ij}}{B_e}$ 
27. Else if  $t_i, t_j$  are transmitted by  $WAN$ 
28.  $D_{tran}(t_i, t_j) = \frac{d_{ij}}{B_c}$ 
29. End If
30. End For
31. End For
32. Calculate  $D,E,R$  according to (7)(11)(14)
33. Return  $M$ 
    
```

In order to conserve good solutions found during the evolution process, we apply fast non-dominated sorting algorithm and crowding distance strategy [13] to select FN individuals from the current population as the population of next generation. The fast non-dominated sorting algorithm is used to generate the non-dominated frontier. The crowding distance measure is used to sort the solutions of the same frontier, and thus remove the redundant solutions.

5 Performance Evaluation

5.1 Experimental Setup

We use an Edgecloudsim [14] simulation tool, to simulate a mobile edge-cloud environment with six mobile terminals, five edge servers and two cloud servers for evaluating the effectiveness of the proposed offloading method. Table 1 shows the parameter values used by the model in Sect. 3. The proposed algorithm is implemented in the Java language. All simulation experiment ran on a personal computer with 2.00 GHz CPU and 8 GB RAM. We apply six real-word workflow application from Pegasus project [15]. Their structural framework are depicted in Fig. 2.

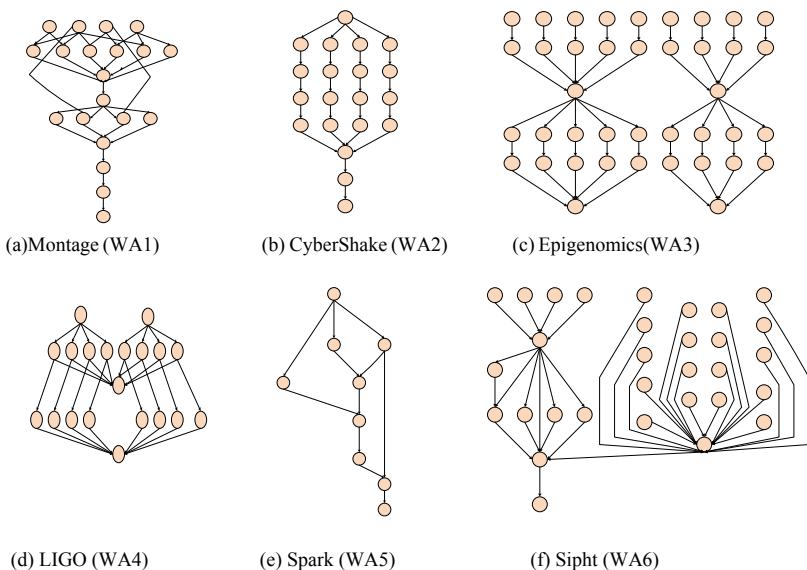


Fig. 2. Structure of six types of workflows.

The proposed offloading method (EAH) is compared with EM [16], ECL [17], EC [5], EAG [7]. EM perform all workflow tasks in the mobile terminal on basis of decisive path scheduling. ECL offload all workflow tasks to edge server on basis of NSGA III algorithm. EC perform all workflow tasks to cloud server on basis of whale

optimization algorithm. EAG perform all workflow tasks in mobile terminal, edge server and cloud server on basis of genetic algorithm, respectively. Parameter of HABC are set to be *population size* = 50, *maximum iterations* = 1000, *limit* = 200, *crossover probability* = 0.8, *mutation probability* = 0.5. The parameter values of NSGA-III are the same as HABC. Parameter of Whale algorithm are set to be *population size* = 50, *maximum iterations* = 1000, *upper bounds* = 4.4, *lower bounds* = 0.5 and *search dimension* = 10.

5.2 Simulation Result

Figure 3 shows the energy consumption of five methods for the different number of workflows tasks. We can see that EM performs the worst in terms of energy consumption, because all tasks are executed on the mobile terminal with limited resource. Compared to EM, EC and ECL perform better in terms of energy consumption of mobile terminal, because some tasks are offloaded on the edge servers and the cloud servers. EAG outperforms EM, EC and ECL in terms of energy consumption of mobile terminal, because all tasks are scattered in mobile terminals, edge servers and cloud servers, which means that more tasks are offloaded from mobile terminal to edge servers and cloud servers. EAH performs the best in terms of energy consumption of mobile terminal, because it apply the more comprehensive algorithm to optimize each objective. As workflow tasks increase, EAH always works best.

Table 1. Model parameter settings

Parameter	Value
The power of mobile devices when CPU is idle state	0.001 w
The power of mobile devices when CPU is active state	0.5 w
The transmission power of mobile devices	0.1 w
The processing capacity of mobile devices	500 MHZ
The processing capacity of edge devices	2000 MH
The processing capacity of cloud	3000 MH
The latency of LAN	1 ms
The latency of WAN	30 ms
The bandwidth of LAN	100 kb/s
The bandwidth of WAN	50 kb/s
The average waiting time of tasks in the servers	20 ms

Figure 4 shows the completion time comparison of EM, EC, ECL, EAG, EAH in the case of different number of workflow tasks. We can find that EAH always performs the best under different scales of tasks. EM takes the most time to perform tasks, because the resources of mobile terminals, and the task execution speed are limited. Compared to EM, EC consumes less time, because the cloud servers has rich resources, large bandwidth and fast processing capacity. Compared to EC, ECL spends less time on execution of tasks. This is because of the smaller delay, and the shorter distance

between the edge servers and mobile terminals. Compared to EC, ECL and EAG, EAH is the fastest to perform tasks, because tasks are separately distributed to mobile terminals, edge servers, and cloud servers so as to effectively reduce waiting time, transmission time and execution time.

Figure 5 shows the reliability comparison of EM, EC, ECL, EAG and EAH in the case of different numbers of workflow tasks. We can find that EAH performs the best in terms of the reliability. The factors affecting the reliability include the calculation cost and the processing capacity of tasks. The higher the calculation cost, the lower the reliability. The stronger the processing capacity of the tasks, the higher the reliability. On the one hand, mobile terminals have low computing costs of tasks but poor processing capacity of tasks. On the other hand, cloud servers have high computing cost but powerful computing capacity. Therefore, both EM and ECL fail to achieve a good tradeoff between the computing costs and the processing capacity, and thus they are not the best offloading solution. Although both EAH and EAG allocate tasks to mobile terminals, edge servers and cloud servers, EAH outperforms EAG in terms of the reliability. The reason is that EAH can search the solution space more efficiently and globally so that it can obtain a better tradeoff between the computing costs and the processing capacity.

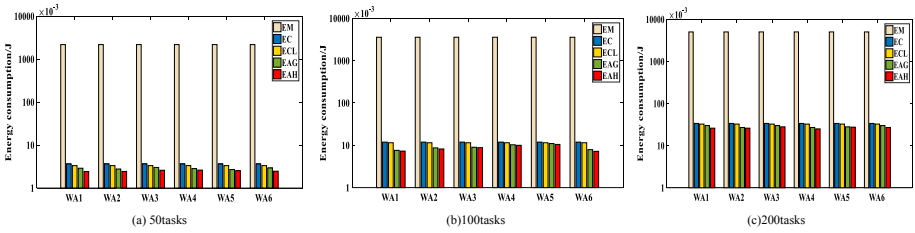


Fig. 3. Energy consumption for six types of workflows under different scales of tasks.

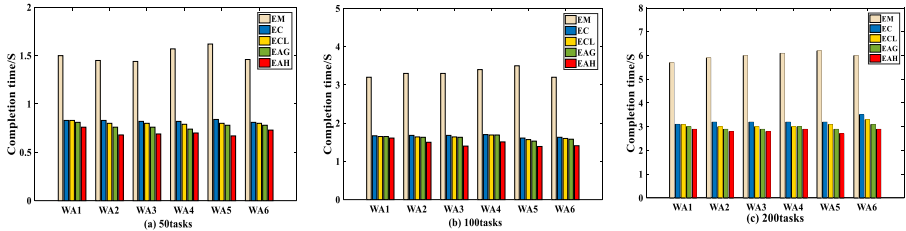


Fig. 4. Completion Time for six types of workflows under different scales of tasks.

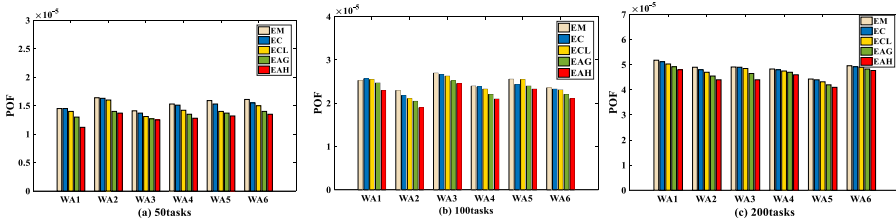


Fig. 5. POF for six types of workflows under different scales of tasks.

6 Conclusion

This paper investigates the problem of workflow tasks offloading in mobile edge-cloud environment. First, we formulate it as a multi-objective optimization model with the goal of optimizing energy consumption of mobile terminal, workflow completion time and workflow reliability, and then we develop a hybrid algorithm involving artificial bee colony, differential evolution algorithm and decoding heuristic, which may find the optimal offloading strategy. Finally, we verify that our method is more energy-saving, time-saving and reliable than the state-of-the-art methods applied to similar problems. In the future, we will consider the impact of the mobility of users on workflow task offloading strategy in mobile edge-cloud computing.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grant 61662052, in part by the Inner Mongolia Science and Technology Innovation Team of Cloud Computing and Software Engineering, in part by the Inner Mongolia Application Technology Research and Development Funding Project, and in part by the Major Research Plan of Inner Mongolia Natural Science Foundation of China under Grant 2019ZD15.

References

1. Wu, H.P., Wu, S.W.: Joint optimization method for task scheduling time and energy consumption in mobile cloud computing environment. *Appl. Soft Comput. J.* **80**(2019), 534–545 (2016)
2. Cui, Y., Song, J., Miao, C.: Mobile cloud computing research progress and trends. *Chin. J. Comput.* **40**(2), 273–295 (2017)
3. Xie, R.C., Huang, T.: *Principle and Practice of Edge Computing*. China Post and Telecommunications Press, Beijing (2019)
4. Shi, W., Cao, J., Zhang, Q.: Edge computing: vision and challenges. *IEEE Internet Things J.* **3**(5), 637–646 (2016)
5. Hu, H.Y., Liu, R.H.: Multi-objective optimization for task scheduling in mobile cloud computing. *J. Comput. Res. Dev.* (2017).
6. Pu, J.: *Research on Task Scheduling of Mobile Cloud Computing Based on DVFS and Heat Perception*. Huazhong University of Science and Technology, Hubei (2016)
7. Fu, S.C., Fu, Z.J.: Computation offloading method for workflow management in mobile edge computing. *J. Comput. Appl.* **39**(5), 1523–1527 (2019)

8. Mao, Y., Zhang, J.: Dynamic computation offloading for mobile-edge computing with energy harvesting devices. *IEEE J. Sel. Areas Commun.* **34**(12), 3590–3605 (2016)
9. Kong, Y.: Research on Task Transfer Strategy in Mobile Edge Computing Environment. Xi'an University of Technology, Xi'an (2018)
10. Vilaplana, J., Solsona, F., Teixidó, I., Mateo, J., Abella, F., Rius, J.: A queuing theory model for cloud computing. *J. Supercomput.* **69**(1), 492–507 (2014). <https://doi.org/10.1007/s11227-014-1177-y>
11. Zhang, L., Li, K., Xu, Y.: Maximizing reliability with energy conservation for parallel task scheduling in a heterogeneous cluster. *Inf. Sci.* **319**, 113–131 (2015)
12. Ma, W., Xun, Z. X.: Artificial bee colony algorithm based on elite swarm search strategy. *J. Comput. Appl.* (2014)
13. Cagatay, S., Atay, O.: EdgeCloudSim: an environment for performance evaluation of edge computing systems. In: International Conference on Fog and Mobile Computing (2017)
14. Deb, K., Agrawal, S., Pratap, A., Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In: Schoenauer, M. (ed.) PPSN 2000. LNCS, vol. 1917, pp. 849–858. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45356-3_83
15. Bharathi, S., Chervenak, A., Deelman, E., Mehta, G., Vahi, K.: Characterization of scientific workflows. In: Proceedings of the The Third Workshop on Workflows in Support of Large-Scale Science, Austin, TX, USA, p. 10 (2008)
16. Peng, Z.R., Wang, G.J.: An optimal energy-saving real-time task-scheduling algorithm for mobile terminals. *Int. J. Distrib. Sens. Netw.* (2017)
17. Peng, K., Huang, H.L., Pan, W.J.: Joint optimisation for time consumption and energy consumption of multi-application and load balancing of cloudlets in mobile edge computing. The Institution of Engineer and Technology (2020)



A Deep Reinforcement Learning Based Feature Selector

Yiran Cheng¹(✉), Kazuhiko Komatsu², Masayuki Sato¹,
and Hiroaki Kobayashi¹

¹ Graduate School of Information Sciences, Tohoku University, Sendai, Japan
`cheng.yiran.t7@dc.tohoku.ac.jp`

² Cyberscience Center, Tohoku University, Sendai, Japan

Abstract. In the field of data mining and machine learning, it is a challenge for researchers and engineers to analyze and classify the high-dimensional data. In order to minimize the classification error, it is critical to identify and select the most characterizing features as well as remove the irrelevant features from the high-dimensional data. Feature selection algorithms provide effective processes to find a compact valuable feature subset from the set of all the candidate features. In past researches, most feature selection methods treat such a problem as a scoring problem, where each feature is evaluated individually and top-ranked features are selected. In this study, adding new features into the feature subset is considered as a Markov Decision Process, and the Feature Selection task is formalized as a reinforcement learning problem. This paper proposes a novel Deep Reinforcement Learning based Feature Selector (DRLFS) along with a dynamic randomness policy and two search protocols to tackle the exploration versus exploitation dilemma for a gradual approach to the optimum subset. The experimental results on various benchmark datasets prove the promising feature selection ability of the proposal of this paper.

Keywords: High-dimensional data · Feature selection · Reinforcement learning · Data mining · AutoML

1 Introduction

With the development of science and technology, modern information technologies and applications generate data at an unprecedented speed like an explosion. To make these data valuable instead of being a burden on a storage system, these data need to be analyzed and classified by machine learning classification algorithms such as Naive Bayes, K-Nearest Neighbor, C4.5 and Support Vector Machine [10, 12, 19]. However, one characteristic of these data is that they are high-dimensional, which means these data contain instances with enormous numbers of features. Most of these features are redundant and irrelevant for prediction, and they will interfere with the analysis onto the data since these

features have no predictive power [8]. As a consequence, lots of computation time will be wasted on the classification due to these data’s enormous size, and the accuracy of the classification algorithm will be harmed, too. In both theory and practice, feature selection methods have been proven valuable and effective for removing redundant and useless features to increase the performance of classification as well as reduce the size of the data [26,27].

Feature selection is usually utilized for the purpose of reducing the dimension of the data, which refers to generating a subset of the original full feature set with the irrelevant features being removed according to a certain feature selection strategy [3]. It works as a preprocess before classifying the data and is widely used in many fields, such as image recognition [7], text mining [24], data mining [15], fault diagnosis [18], and so on. The obtained feature subset will be a better representative of the original data since it reduces the test error and increases the effectiveness of classification algorithms. Compared with selecting features by human efforts, feature selection does not require a deep comprehension of the data or rich experiences in handling high-dimensional data.

Formerly, there were several rule-based heuristic and machine learning-based non-heuristic feature selection algorithms. However, most of the conventional rule-based approaches have not considered the inter-connections between features, which will end up with the disability of finding the optimal feature subset [6]. On the other hand, search policies of non-heuristic approaches will also limit the exploration. Thus, this paper proposes a Deep Reinforcement Learning based Feature Selector (DRLFS) by leveraging the state-of-the-art reinforcement learning [22] algorithm to automatically remove redundant features for the purpose of improving the accuracy. During the feature selection process, the agent automates the feature selection process by learning-based policies instead of relying on experienced experts or heuristic algorithms. To utilize the guidance ability of Exploitation and retain the searching power of Exploration at the same time, a dynamic randomness policy and two search protocols are also proposed in this work.

The rest of this paper is organized as follows. Section 2 discusses related work and challenges of performing feature selection over high-dimensional data. Section 3 then describes the detailed information of the proposed Deep Reinforcement Learning based Feature Selector. In Sect. 4, the proposed feature selector is evaluated over various kinds of benchmark or competition datasets. Finally, the last section gives conclusions of this paper and describes the future work.

2 Related Work

Feature selection mainly aims to improve the accuracy of the learned hypothesis by removing noisy redundant features from high-dimensional data. Formerly, there exist three main categories of feature selection algorithms: filter-based approaches, wrapper-based approaches, and embedded approaches.

Filter-based approaches simply score features independently by certain scoring functions, and then select top-ranked features. These scoring functions value

the correlation between the feature and the class value. An example is CFS [9], which evaluates the effect of every single feature corresponding to each class to obtain the optimal subset of the features set finally. FCBF (Fast Correlation Based Filter) extends the CFS by a faster implementation, and enables FCBF to select any given size of feature subset [21]. RELIEF and its improved algorithm RELIEFF (modified to support multi-class problem) evaluate each the influence of each feature onto each example neighborhood (the Euclidean distance built from all features) [13,14]. The claw of filtering approaches is that they ignore the interdependencies between different features.

Embedded approaches are featured by combining learning process and feature selection through the prior or posterior regularization. An approach select subsets by minimizing the residual sum of L_1 norm [23], which also inspires the approach named Least Absolute Shrinkage and Selection Operator (Lasso) [4]. Another approach performs hierarchical multiple kernel learning (HKL) and select among kernels by L_1 type penalties [1]. Other approaches use non-linear hypotheses or random forests to compute feature scores [2,20,25]. These embedded approaches share a common problem that the feature redundancy hinders their scores in a feature selection perspective.

Wrapper approaches explore the powersets of the whole original features and measure the performance (the generation error) of all considered subsets from all combinations [6]. It is critical to navigate the agent to reach the optimal combination and solve the Exploration versus Exploitation dilemma in such a kind of approach. Michalak et al. proposed a relationship-based dual strategies approach to find the optimal feature subset [17]. As one of the non-heuristic model-based reinforcement learning methods, the Feature UCT SElection (FUSE) algorithm proposed by Romaric Gaudel extends the Monte-Carlo tree search Upper Confidence Tree to settle the Exploration versus Exploitation dilemma and deal with the finite unknown horizon [6]. Moreover, Fard et al. advances this algorithm by an improved Upper Confidence Graph and generate a wrapper-based Feature Selector in his Feature Selection using Temporal Difference (FSTD) method [5]. Such two methods are constrained by their weak exploration ability of the searching model. Wrapper-based approaches are considered the best because of their abilities to find out the optimum subset. However, these approaches also suffer from the problem of their high time complexity [5].

In summary, both the filtering and embedded approaches try to select features based on an evaluation of every feature: either evaluate separately, which causes the lost consideration of correlations between different features, or evaluate together with all other features, which is interfered with feature redundancy. Only the wrapper-based approaches are based on generating various combinations of feature subsets and have a chance to approach the optimum. The cost of such ability is that all wrapper-based approaches must be evaluated on sufficient number of subsets, which will be costly on time [6]. Moreover, the Exploration vs. Exploitation dilemma that obstructs reaching the optimum subset is the problem that every wrapper approach must resolve. Since the method proposed in this work is a wrapper-based, a problem must be solved while searching over

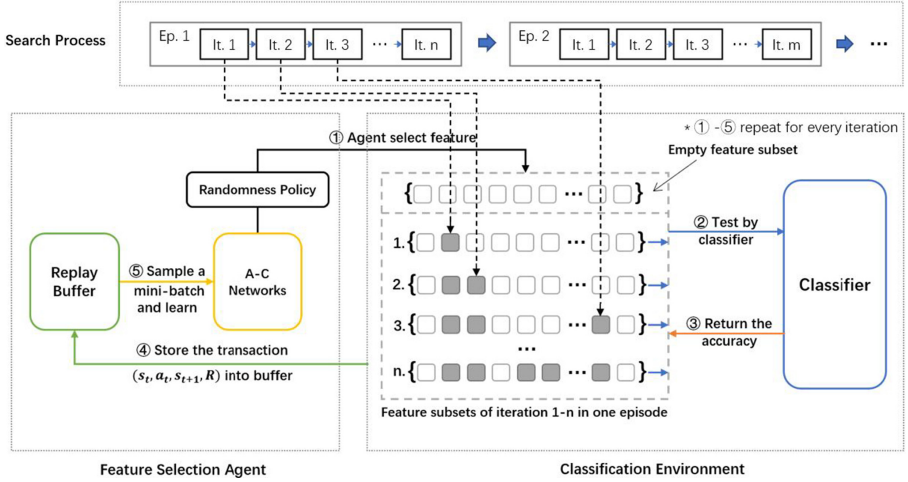


Fig. 1. Overall structure of DRLFS, the procedure from ① to ⑤ repeats for every iteration 1 to n.

the feature subset space: The Exploration versus Exploitation tradeoff should be controlled efficiently to gradually narrow the search space of feature subsets for the sake of scalability. With the problem being tackled, the optimum feature subset would be gradually dug out.

3 Deep Reinforcement Learning Based Feature Selector

3.1 Overview

Model-free reinforcement learning algorithms always excel at searching over the unknown action space based on their trying and learning mechanism. Thus, the Deep Reinforcement Learning based Feature Selector (DRLFS) is proposed here to find the optimum feature subset. Figure 1 shows the overall structure of the proposed DRLFS. The DRLFS embodies a reinforcement learning based agent and an interactive environment. The agent provides a learning-based policy to select features in the feature subset space and learn from transactions, while the environment is composed of a classifier and a data modifier to select features based on actions and calculate the reward. In order to efficiently search the feature subset space, a randomness policy and two search protocols are proposed to control the Exploration versus Exploitation tradeoff.

The whole searching procedure of the feature selection agent is divided separately into episodes. Every episode is a consecutive sequential process to continuously select features into the feature subset until the agent reaches the terminate state. Moreover, each episode consists of multiple iterations. For every iteration, the agent selects one feature into the feature subset, tests the accuracy, calculates the reward, stores the transaction and learns from a sampled mini-batch.

Specifically, for every iteration, the agent receives an embedding state s_t from the environment, and then generates a decision of selected feature as an action a_t by the learned policy of reinforcement learning as well as the randomness policy proposed in this paper. The feature subset adds the selected feature by a_t . Then, the environment modifies the data by dropping un-selected features and classifies the data with remaining features using the specified classification algorithm. Finally, the evaluation of the accuracy is performed on the validation set, and the reward is calculated to feed the agent for learning. In every episode, former process repeats until current episode is terminated by the search protocol introduced later.

3.2 Feature Selection as a Sequential Reinforcement Learning Problem

This paper formulates the feature selection task as a sequential reinforcement learning problem. Under this concept, here the detailed setting of such reinforcement learning problem is introduced. Each transition in an iteration is (s_t, a_t, s_{t+1}, R) , where s_t is the state, a_t is the action, and R is the reward collected by the environment after evaluating the accuracy performance of the coordinate feature subset provided by the action.

Let \mathcal{F} denote the set of all available features, the state space \mathcal{S} (a finite set of states) of the Markov Decision Policy (MDP) is composed of powersets of \mathcal{F} . Since the selection of feature is a binary decision, it could be derived that $|\mathcal{S}| = 2^{|\mathcal{F}|}$. As for action space \mathcal{A} , much evidence shows that a slight change in action may eliminate some key features and break the inter-dependencies between features [26]. Thus, every feature needs to be configured in the action space as a representative, which leads to an explosion of the number of discrete actions. Such a large action space is difficult to explore efficiently. Also, since the classifier's performance is highly sensitive to the decision on every feature, it is not recommended to downsample the feature space to group features up. Thus, a fine-grained continuous action space $\mathcal{A} = (0, 1]$ is used here to enable a fine-grained action space as output. Let $d = |\mathcal{F}|$ denote the number of dimensions of the data. The selected feature f in iteration t is equal to $[d \cdot a_t] - 1$. For each episode, the feature selection agent starts with an empty subset F , and in each iteration of the episode, it sequentially selects a feature $f \in \mathcal{F}$ by learned deterministic policy $\mu: \mathcal{S} \rightarrow \mathcal{A}$, add the feature f to the subset F , until it reaches the terminate condition when the feature subset becomes the final subset F_e .

Let $Error(\mathcal{F})$ and $Error(F_e)$ denote the test error of learned hypothesis trained on all features \mathcal{F} and that of learned hypothesis trained on features in the final subset F_e , respectively. Since the ultimate objective of feature selection is to generate a best representative subset of \mathcal{F} that yields the minimal test error at the end of the learning process, the reward function should be:

$$R = Error(\mathcal{F}) - Error(F_e) \quad (1)$$

Clearly the optimal deterministic policy μ^* is the one that could maximize the reward:

$$\mu^* = \arg \max_{\mu} \text{Error}(\mathcal{F}) - \text{Error}(F_e) \quad (2)$$

To generate the optimal policy, the proposed feature selection agent leverages the Deep Deterministic Policy Gradient (DDPG), an off-policy actor-critic reinforcement learning algorithm in order to learn through trial and error, as well as control the fine-grained action over the feature subset consecutively and precisely [16]. The actor-critic structure of DDPG also helps reduce the variance and facilitate a stabler training [11].

3.3 Randomness Policy

As discussed formerly, the way to deal with Exploration versus Exploitation dilemma is critical to navigating the agent in the feature powerset lattice. Exploration refers to the random choice of features for the purpose of exploring more possibilities. Exploitation is the tendency to choose the best feature set that has been explored formally to gain the highest reward. DDPG is an off-policy algorithm, which means the problem of exploration is able to be treated independently from the learning algorithm [16]. Generally, the exploration policy μ' is achieved by adding noise collected from a specific noise policy \mathcal{N} to the agent's actor policy:

$$\mu'(s_t) = \mu(s_t | \theta_t^\mu) + \mathcal{N} \quad (3)$$

The Gaussian policy, an exploration noise policy that generally used for randomizing the continuous action, causes problems such as repeatedly exploration and insufficient exploitation in complex feature selection tasks due to its naive mechanism. To tackle such problems, a dynamic randomness policy is proposed here to randomize the selected action a_t generated based on state s_t of episode t . The truncated normal distribution with lower bound 0 and upper bound $d-1$ is used as the exploration noise. Then the actor policy is provided by the following equation:

$$\mu'(s_t) \sim \text{TN}(\mu(s_t | \theta_t^\mu), \sigma^2, 0, d-1), \quad (4)$$

where $\mu(s_t | \theta_t^\mu)$ indicates the output of the actor network and $\mu'(s_t)$ indicates the actual output under the randomness policy. Then the action is selected by $a_t = \mu'(s_t)$. The standard deviation σ is set by the proposed strategy called the decaying dynamic standard deviation to confront the Exploration vs. Exploitation dilemma: Assume the feature selection agent reaches iteration t of episode i and the state becomes s_t . Then, the value of σ is given by the following equation:

$$\sigma = \begin{cases} \min(c^t \sigma'_i, d) & \text{if } s_t \text{ has been explored} \\ \sigma'_i & \text{otherwise,} \end{cases} \quad (5)$$

where σ'_i is the starting standard deviation that provides the starting value of σ in each episode and $c \in (1, 10]$ is an impelling hyperparameter to enhance exploration if current search space has been reached by the agent formerly. Equation (5) means in one episode, when current state has been reached, the value of

Algorithm 1: DRLFS

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$
Initialize target network Q' and μ'
Initialize the Replay Buffer R
for $episode = 0$ to N **do**
 Initialize the feature subset F as an empty set
 Initialize the state s_0 as a zero vector with length d
 while s_t is not s_e **do**
 According to the current actor policy, select the action $a_t = \mu(s_t|\theta^\mu)$
 Randomize the action a_t by truncated normal distribution and decaying dynamic standard deviation
 Transform s_t to s_{t+1} and add newly selected feature into F by a_t
 Test generalization error on classifier and calculate the reward r_t
 Store transaction (s_t, a_t, r_t, s_{t+1}) into R
 Sample a random minibatch from R
 Update critic $Q(s, a|\theta^Q)$ by minimizing the loss
 Update the actor policy $\mu(s|\theta^\mu)$ using the sampled policy gradient
 Update the target networks Q' and μ'
 if # features = limit **then**
 Set $s_t = s_e$
 end if
 end while
end for

σ is multiplied by c , and once the agent reaches an unexplored search space, the value of σ is fixed to σ'_i in the rest of iterations of this episode.

As for the value of σ'_i , to approximate the even distribution, the value of σ'_i is set to d at the early stage of the whole training process. Let r denote the episode-wise decay factor. After the agent has been trained with half of the max episodes, in order to decay the randomness episode-wisely for exploitation, the value of σ'_i decreases along with the increment of episode i by the following equation:

$$\sigma'_i = \begin{cases} d & \text{if } i < I/2 \\ d \cdot r^{i-I/2} & \text{otherwise,} \end{cases} \quad (6)$$

where I indicates the number of total episodes to train the agent. The episode-wise decaying factor r is derived by $\frac{1}{1/\sqrt[2]{2d}}$.

This randomness policy gives a way to avoid repeated exploration of the search space, motivates the agent to search unexplored space, and enhances the exploration in the late stage while exploitation dominants. As a result, the agent is able to search more efficiently under the proposed randomness policy.

3.4 Search Protocols

In the overall searching process of DRLFS that illustrated in Algorithm 1, each episode requires a criterion to terminate current episode and start over another episode. Define the number of selected features in the feature subset as the

search depth, here two search strategies are used to constrain the search depth by limiting the number of selected features.

Fixed Depth Search simply limits the size of the feature subset that the agent can reach. In the fixed depth search, the search process in each episode stops at a fixed depth. Under this fixed depth search, the stopping state for the agent will be $s_e \in \{s \mid \#features = l\}$, where l is a hyper-parameter that sets a limitation of max depth that the agent can reach. Such policy provides a stable search evenly for every depth, which means that all different depths are explored for the same number of times.

Adaptive Depth Search provides a policy in which the limit of max depth gradually increases by estimating whether the search space of the current depth is fully explored. Let $Error'_n$ denote the optimum error rate of depth n (or feature subset with n features). Define the memorize set $E = \{Error'_n \mid 0 < n \leq d\}$. Let l' denote the depth limitation. As same as the fixed depth search, the stopping state is given by $s_e \in \{s \mid \#features = l'\}$. Here, the value of l' starts with 1 at the beginning of training, and will increase by 1 if E is unchanged for 200 episodes. Comparing to the fixed depth search, the adaptive depth search tends to fully explore the current search space before the agent moves to a deeper search space.

Table 1. Properties of used datasets

No	Dataset	Features	Instances
1	Arcene	10000	900
2	Spambase	4601	57
3	Colon	2000	62

4 Performance Evaluation

The goal of the experiment is to answer two main questions: the overall performance of the DRLFS-FD (DRLFS with Fixed Depth Search) and the DRLFS-AD (DRLFS with Adaptive Depth Search), and the comparative strengths as well as weaknesses of the two proposed feature selection methods.

4.1 Experimental Setting

Table 1 shows the properties of the Arcene, Spambase and Colon datasets of the UCI benchmark datasets used in this work to evaluate the proposed feature selector. Due to search space limitations, they are all binary classification problems,

and numerical features are considered. The objective of Arcene dataset is to distinguish cancer versus non-cancer patterns from the data of mass-spectrometric. It contains 10,000 features with 900 instances, including 7,000 real features and 3,000 distractor features with no predictive power. The Colon dataset contains 62 samples with information on 2000 genes that belong to the tumor and normal colon tissues. Spambase is a dataset with 4601 instances on 57 correlated features, and the target is to distinguish whether an e-mail is a spam. All of these datasets are widely used in feature selection challenges.

For the sake of the robustness and a fair comparison of the proposal with other state-of-the-art feature selection approaches, all of the approaches are tested with a same classifier, the Gaussian SVM. Moreover, all reported test results are averaged over 5 independent runs. In each run, a 10-fold stratified cross validation is performed. For each fold, DRLFS is trained with 20000 episodes at most.

Comparative experiments are conducted between the CFS (correlation-based feature selection) [21], the Gini-RF (random forest based Gini score) [2], FUSE (Feature Uct Selection) [6], FSTD (Feature Selection Using Temporal Difference) [5] and proposals of this work, DRLFS-AD and DRLFS-FD. The baselines are test accuracy of the mentioned predictor SVM in the absence of feature selection procedures, which means all the features of each dataset are used for classification. Moreover, the Gaussian policy based DRLFS-FD agent (noted as DRLFS*) is also deployed to compare the generally used Gaussian policy with the proposed dynamic randomness policy.

4.2 Evaluation Results and Analysis

Figures 2 (a), (b), and (c) give results of experiments for comparison among feature selection methods on Colon, Spambase, and Arcene respectively. The vertical axis indicates the accuracy on test data while the horizontal axis indicates the number of selected features.

First, it can be deduced from all the figures that the influence of the feature selection approaches is significant for not only alleviating calculation stress by reducing the number of features, but also improving the performance of the classification as well while the proper feature subset is found. The regions over the baseline in these figures indicate that after the feature selection process, with redundant and irrelevant features removed, the accuracy improves significantly. These figures prove that all the feature selection approaches are able to generate efficient feature subsets, which raise the accuracy and reduce the data size at the same time.

Compared with the other approaches, the proposed DRLFS-AD and DRLFS-FD take lead in all the experiments. Especially in the experiments on the Colon dataset, two proposed ones outperform the other approaches with 10% to 20% of accuracy. On the one hand, the proposed ones could filter out irrelevant features, thus, they perform well when considering few features. On the other hand, they are able to select features with inter-dependencies so that they have better performance while considering many features. To be noticed, two conven-

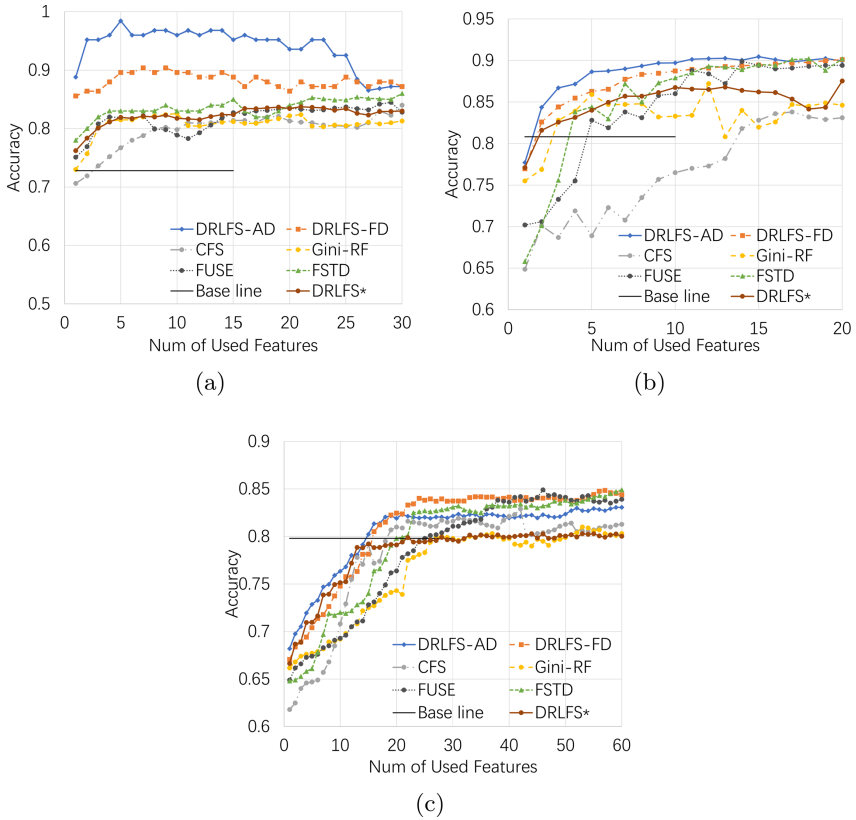


Fig. 2. Accuracy versus the number of selected features on (a) Colon, (b) Spambase, and (c) Arcene.

tional feature selection approaches, the filter-based CFS and embedded Gini-RF, obtain lower accuracy than all the wrapper-based approaches. Moreover, compared to two machine learning based approaches, FSTD and FUSE, DRLFS-AD and DRLFS-FD also show better results. This is because the randomness policy used in DRLFS is able to control the Exploration versus Exploitation dilemma in an efficient way.

Comparison between DRLFS-AD and DRLFS-FD is also illustrated in Fig. 2. DRLFS-AD tends to be more efficient in searching over the search space with fewer features and shows a better performance from 1 up to about 20 used features. On the contrary, DRLFS-FD is able to provide a better result in the cases that the number of considered features exceeds 20. This is because the search policy of DRLFS-AD tends to spend more exploration on the search space with fewer features and spend more exploitation in the late stage while the number of features becomes larger. Moreover, the impelling factor fails to compensate the fully decayed starting standard deviation at late stage while

searching the feature space with more features. On the contrary, DRLFS-FD provides a stable search evenly for the feature space with the different number of features.

Furthermore, compared to two dynamic randomness policy based approaches, the accuracy of the Gaussian policy based DRLFS (DRLFS*) is not satisfying. DRLFS-AD and DRLFS-FD outperform DRLFS* with 5% up to 10% of accuracy in most cases, which proves that the proposed dynamic randomness policy is able to provide a more efficient search over the feature subset space. This is because of the repeated exploration during the early stage of the whole searching process as well as insufficient exploitation during the late stage of the Gaussian policy.

5 Conclusions and Future Work

Conventional feature selection approaches using rule-based policies to explore a large search space of feature subset is usually suboptimal due to the neglect of inter-dependencies between different features. In this paper, the task of feature selection is formalized as a sequential reinforcement learning problem. In addition, the state-of-the-art reinforcement learning algorithm is leveraged to output a continuous valued action space for a precise control over the feature subset space. Furthermore, to solve the Exploration versus Exploitation dilemma in an efficient way, a dynamic randomness policy and two search protocols, the fixed depth search and the adaptive depth search, have been proposed. The results of the comparative experiments give some solid evidence that the proposed DRLFS-AD and DRLFS-FD achieve higher accuracy than the conventional approaches on classification tasks.

The experimental results show that DRLFS-AD performs better when the number of features is lower than 20, and DRLFS-FD obtains better accuracy in the cases that the number of features is more than 20. To make the proposal work well on both cases, the future work would focus on combining the strength of two search protocols.

References

1. Bach, F.: Exploring large feature spaces with hierarchical multiple kernel learning. *Adv. Neural Inf. Process. Syste.* **21**, 105–112 (2008)
2. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees. belmont, ca: Wadsworth. International Group **432**, 151–166 (1984)
3. Cai, J., Luo, J., Wang, S., Yang, S.: Feature selection in machine learning: a new perspective. *Neurocomputing* **300**, 70–79 (2018)
4. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al.: Least angle regression. *Annal. Stat.* **32**(2), 407–499 (2004)
5. Fard, S.M.H., Hamzeh, A., Hashemi, S.: Using reinforcement learning to find an optimal set of features. *Comput. Math. Appl.* **66**(10), 1892–1904 (2013)
6. Gaudel, R., Sebag, M.: Feature selection as a one-player game (2010)

7. Goltsev, A., Gritsenko, V.: Investigation of efficient features for image recognition by neural networks. *Neural Networks* **28**, 15–23 (2012)
8. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *J. Mach. Learn. Res.* **3**, 1157–1182 (2003)
9. Hall, M.A.: Correlation-based feature selection of discrete and numeric class machine learning (2000)
10. Han, J., Pei, J., Kamber, M.: *Data mining: concepts and techniques*. Elsevier (2011)
11. He, Y., Lin, J., Liu, Z., Wang, H., Li, L.-J., Han, S.: AMC: AutoML for model compression and acceleration on mobile devices. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) *ECCV 2018*. LNCS, vol. 11211, pp. 815–832. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_48
12. Huang, D., Du, J.: A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks. *IEEE Trans. Neural Networks* **19**(12), 2099–2115 (2008). <https://doi.org/10.1109/TNN.2008.2004370>
13. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: *Machine Learning Proceedings 1992*, pp. 249–256. Elsevier (1992)
14. Kononenko, I.: Estimating attributes: analysis and extensions of RELIEF. In: Bergadano, F., De Raedt, L. (eds.) *ECML 1994*. LNCS, vol. 784, pp. 171–182. Springer, Heidelberg (1994). https://doi.org/10.1007/3-540-57868-4_57
15. Li, J., Liu, H.: Challenges of feature selection for big data analytics. *IEEE Intell. Syst.* **32**(2), 9–15 (2017)
16. Lillicrap, T.P., et al.: Continuous control with deep reinforcement learning. arXiv preprint [arXiv:1509.02971](https://arxiv.org/abs/1509.02971) (2015)
17. Michalak, K., Kwasnicka, H.: Correlation-based feature selection strategy in neural classification. In: *Sixth International Conference on Intelligent Systems Design and Applications*, vol. 1, pp. 741–746. IEEE (2006)
18. Rauber, T.W., de Assis Boldt, F., Varejão, F.M.: Heterogeneous feature models and feature selection applied to bearing fault diagnosis. *IEEE Trans. Industr. Electron.* **62**(1), 637–646 (2014)
19. Rodriguez, A., Laio, A.: Clustering by fast search and find of density peaks. *Science* **344**(6191), 1492–1496 (2014). <https://doi.org/10.1126/science.1242072>, <https://science.sciencemag.org/content/344/6191/1492>
20. Rogers, J., Gunn, S.: Identifying feature relevance using a random forest, pp. 173–184, January 2005. https://doi.org/10.1007/11752790_12
21. Senliol, B., Gulgezen, G., Yu, L., Cataltepe, Z.: Fast correlation based filter (FCBF) with a different search strategy. In: *2008 23rd International Symposium on Computer and Information Sciences*, pp. 1–4. IEEE (2008)
22. Sutton, R.S., Barto, A.G.: *Reinforcement Learning: An Introduction*. MIT Press, Cambridge (2018)
23. Tibshirani, R.: Regression shrinkage and selection via the lasso. *J. Roy. Stat. Soc.: Ser. B (Methodol.)* **58**(1), 267–288 (1996)
24. Van Landeghem, S., Abeel, T., Saeys, Y., Van de Peer, Y.: Discriminative and informative features for biomolecular text mining with ensemble feature selection. *Bioinformatics* **26**(18), i554–i560 (2010)
25. Varma, M., Babu, B.R.: More generality in efficient multiple kernel learning. In: *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 1065–1072 (2009)
26. Zhang, R., Nie, F., Li, X., Wei, X.: Feature selection with multi-view data: a survey. *Inf. Fusion* **50**, 158–167 (2019)
27. Zhao, Z., Morstatter, F., Sharma, S., Alelyani, S., Anand, A., Liu, H.: Advancing feature selection research. ASU feature selection repository, pp. 1–28 (2010)



Automatic Thread Block Size Selection Strategy in GPU Parallel Code Generation

Weifang Hu, Lin Han, Pu Han^(✉), and Jiandong Shang

Supercomputing Center of Henan, Zhengzhou University, Zhengzhou, China
wfhu@gs.zzu.edu.cn, hanpu2008@126.com

Abstract. In order to improve the flexibility of thread configuration during GPU parallel code automatic generation, an automatic selection strategy of thread block size based on the occupancy of multiprocessors is proposed, and an analysis method based on the polyhedral model is employed to analyze array accesses of GPU kernels. This strategy evaluates the occupancy of device kernels under different thread block sizes, and selects the thread block size with the highest occupancy. For kernels with complex array access or synchronization statements, it automatically selects smaller thread block size to increase schedule spaces of thread warps. It is implemented in the Polly-Acc module of the LLVM compilation framework, and tested on a Tesla architecture GPU using the PolyBench test set. Compared with using fixed block sizes of 32×8 and 32×16 , this strategy makes the generated code achieved an average performance improvement of 9.7% and 15.5% respectively. At the same time, compared with using the thread block selection suggestions in the CUDA API, an average performance improvement of 21% is obtained.

Keywords: GPGPU · Polyhedral compilation · Auto-parallelization · CUDA

1 Introduction

With the rapid development of heterogeneous computing, the transplantation and deployment of applications for CPU-GPU heterogeneous systems have presented huge challenges. As a fascinating area of research in recent years, GPU code automatic generation technology has become an important component of software stacks such as deep learning compilation frameworks [1–3]. At present, there have been a variety of related code generation tools, such as C-to-CUDA [4], CUDA-Chill [5] and PPCG [6]. These tools can transform source level programs into parallel code that can be compiled and executed on GPU platform, and greatly improve the efficiency of program transplantation and optimization for the GPU platform.

Compiler optimization technology based on polyhedral model plays an important role in automatic generation of GPU-oriented parallel code [7]. Baskaran et al. proposed C-to-CUDA, a polyhedral compiler for GPU platform, and employed Pluto algorithm [8] to explore parallelism and locality. Leung et al. proposed a GPU parallel code generation tool based on the R-Stream compiler [9], which makes full use of the memory hierarchy of GPU platform. The PPCG proposed by Verdoolaege et al. can optimize multi-level parallelism and locality, and generate code that includes device

memory applications and copies. Tobias [10, 11] et al. implemented lower-level code automatic generation in LLVM compilation framework, which manage the data transmission between host and device more precisely. On the other hand, Par4All [12] proposed by Amini et al. uses abstract interpretation of array areas and performs powerful interprocedural analysis of input code instead of employing the polyhedral model.

However, users often need to specify the thread block size of kernels when using these tools for generate GPU code automatically. For computing tasks on multiprocessors, more thread loads will bring more instruction scheduling space, improve instruction-level parallelism, and thus increase the occupancy of multiprocessors [13]. On the contrary, due to the limited register and shared memory resources in multiprocessors, too many thread loads may result in lower performance. In addition, because the best choice of thread block sizes for different kernels in the same program may not be consistent, choosing same thread block size for different kernels may cause performance loss for some kernels.

In order to solve this problem, this paper proposes a selection strategy for determining the optimal thread block size configuration at compile time, and integrates it into the LLVM compilation framework. Based on the usage of registers, shared memory and other resources in kernels, this strategy analyzes the occupancy with different thread block sizes, and selects appropriate block sizes under the guidance of occupancy. If there are complex array access or synchronization statements in kernels, this strategy will choose a smaller thread block size. This ensures that kernels have sufficient parallelism while avoiding the memory access delay caused by the storage resource limitation of multiprocessor. We select some test cases in PolyBench for testing on a GPU platform of Tesla architecture. Compared with using the thread block selection suggestions in the CUDA API, an average performance improvement of 21% is obtained.

2 Related Theory

2.1 CUDA Execution Model

This paper focuses on NVIDIA GPU and its corresponding SIMT execution model [14]. The GPU architecture is composed of a group of multiprocessors that realize the hardware parallelism of GPU. Each multiprocessor contains multiple scalar computing units and an instruction unit, which support concurrent execution for hundreds of threads. When a kernel function grid is started, multiple thread blocks are simultaneously mapped to multiprocessors. Threads in a multiprocessor form a warp according to a certain scale, which is the smallest unit of scheduling. In each multiprocessor, multiple stream processors execute the same instructions in the same clock cycle, but process different data. Each stream processor can process multiple instructions into a pipeline to take advantage of instruction level parallelism.

As the smallest unit of kernel resource allocation, thread block consists of several warps. The resources of each thread block are restricted to the same multiprocessor and are allocated before the kernel starts. This mechanism ensures that all the required

resources are allocated before threads running, enables efficient switching between threads and thus more hiding of instruction delays. Meanwhile, the delay of instructions can be hidden in this way. The completeness of the resources in thread block and the independence between thread blocks make the scheduling of different thread blocks extremely flexible, which improves the scalability and adaptability of programs.

2.2 GPU Code Generation Based on Polyhedral Model

The polyhedral model is used to extract, analyze and transform the SCoP (Static Control Parts) in programs. The SCoP is a collection of consecutive statements whose loops and conditional expressions are affine functions of loop iteration variables and parameters. For the SCoP that conforms to the constraint condition, the iteration domain of each polyhedral statement and the dependencies between statement execution instances can be analyzed. By mapping each element in the iteration domain to a timestamp, the execution order between statement instances can be obtained [15]. This mapping relationship is called the schedule of statement instances.

Taking PLuTo algorithm as an example, a schedule function sequence is calculated for each polyhedral statement. The scheduling function is modeled as $\phi_S = \mathbf{i}\vec{c} + \mathbf{p}\vec{d} + C$. Where \mathbf{i} and \mathbf{p} are vectors composed of iterative variables and parameters in SCoP. \vec{c} and \vec{d} are integer vectors, and C is an integer value. If there is a dependence between statements S and R , where S is the source node and R is the sink node, the schedule conforms to this dependence when schedule function satisfies $\phi_R(\mathbf{i}, \mathbf{p}) - \phi_S(\mathbf{i}, \mathbf{p}) \leq 0$. At this time we can use Farkas lemma to convert dependencies into constraints that the schedule coefficient \vec{c} needs to meet. Then we can construct an integer linear programming problem to solve \vec{c} and get a scheduling function that preserves program semantics.

On the other hand, it is possible to gain data reuse from dependence when the dependence distance is small enough. By applying Farkas's lemma, the upper bound of the dependence distance can be obtained, that is, $\phi_R(\mathbf{i}, \mathbf{p}) - \phi_S(\mathbf{i}, \mathbf{p}) \leq \mathbf{u}\vec{p} + w$, where \mathbf{u} is a row vector. And by solving a lexicographic minimization problem, a schedule that conforms to program semantics and minimizes the distance dependence can be obtained.

For the given constraints, the scheduling algorithm iteratively solves the scheduling function and ensures that the scheduling obtained under the same constraints is linearly independent. When multiple scheduling functions are solved using the same constraints, the sequence of these scheduling functions constitutes a permutable band. That is, because the scheduling functions in each dimension of the band conform to the same dependencies, they can interchange positions arbitrarily. If parallelism constraint is added to the linear programming problem, the schedule of corresponding dimension can be parallelized. The algorithm removes the dependencies that have been satisfied in the current schedule and continues to calculate the next band when the solution algorithm fails to obtain an effective solution.

After completing the schedule calculation, a schedule tree composed of bands [16] will be obtained. The outermost scheduling band that contains at least one parallel loop and nodes below it will be mapped to the GPU to form a kernel function, while the

loops corresponding to the upper schedule band are going to run on the host. Within each kernel, the parallel loops will be split into a pair of loops. Depending on the number of parallel loops, the outer layer 1 to 2 loops are mapped to thread blocks in the grid, and the inner 1 to 3 loops are mapped to threads in the thread block. In this process, the size of blocks depends on the user-specified thread block size, which is typically 32×16 by default. At the same time, since the memory of GPU hardware is separated from the host device, the code generation of memory space application and data copy for the data accessed in this region are required after determine the kernel region. In PPCG, when the thread marking is completed, the array reference groups in the kernel will be analyzed. If multiple threads share data elements in a certain area, the data will be moved to shared memory in advance to improve memory access performance.

3 Selection Strategy for Thread Block Sizes

A GPU is a throughput-oriented system that uses massive parallelism to hide latency. Occupancy is a measure of thread parallelism in GPU kernels. The higher occupancy of multiprocessor, the greater the opportunity for the GPU to achieve higher parallelism and throughput. In GPU code generation based on polyhedral model, there are always many control flows and synchronization statements in programs due to the need to ensure the legitimacy and effectiveness of automatic transformation. In order to ensure that there are enough warps on multiple processors to hide instruction delays, the automatic selection strategy of thread block size proposed in this paper is based on the following principles: on the basis of using the thread block size with the highest occupancy, the schedule space of warps will be increased as much as possible by reducing the thread block size if necessary.

3.1 Improve the Occupancy

The occupancy of a multiprocessor is the ratio of active warps to the maximum active warps supported. Low occupancy results in low instruction throughput because there are not enough warps to hide the delay between instructions [17]. When the number of threads in each thread block is fixed to a certain value, the occupancy of kernels on multiprocessor is limited by register and shared memory resources. The theoretical occupancy can be calculated by the following formula:

$$O = \frac{\min(R_{limit}, S_{limit}, W_P)}{W_P} \quad (1)$$

Where R_{limit} , S_{limit} are the maximum number of active warps on multiprocessor under the limit of register and shared memory resource usage respectively. W_P is the maximum number of active warps supported by hardware.

R and S , which are the number of registers and shared memory used in a kernel, are fixed when GPU code is automatically generated based on the polyhedral model. If the thread block size is $size$, there are:

$$R_{limit} = \left\lfloor \left[\frac{limitR}{\left\lceil R * \frac{W}{R_{size}} \right\rceil * R_{size} * R_g} \right] * R_g / \frac{size}{W} \right\rfloor * \frac{size}{W} \tag{2}$$

$$S_{limit} = \left\lfloor \frac{Smem}{\left\lceil (S + S_{rt}) / S_{size} \right\rceil * S_{size}} \right\rfloor * \frac{size}{W} \tag{3}$$

Where $LimitR$ is the maximum number of registers used for a single thread block. W is the number of threads in a warp. R_{size}, R_g are register application unit size and granularity respectively. $Smem$ is the maximum number of shared memory can be used by a single thread block. S_{rt}, S_{size} are the shared memory size required by the runtime library and the size of shared memory application unit respectively.

After the kernel code is generated and embedded into the host code, the theoretical occupancy of multiple thread block sizes is calculated according to the registers and shared memory usage, and the thread block size with the highest occupancy rate is used for the kernel startup configuration. Figure 1 shows the thread block size selection process guided by occupancy in the LLVM compilation framework.

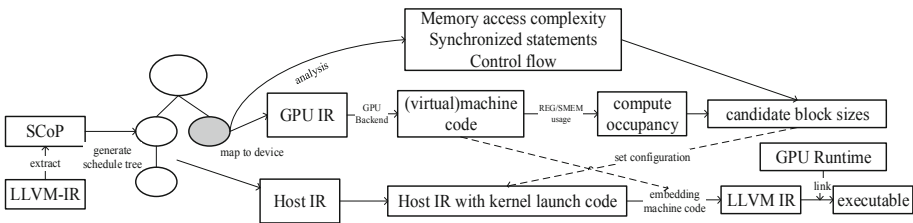


Fig. 1. Block size selection based on occupancy in the LLVM compilation framework.

After the kernel code is generated and embedded into the host code, the theoretical occupancy of multiple thread block sizes is calculated according to the registers and shared memory usage, and the thread block size with the highest occupancy rate is used for the kernel startup configuration. Figure 1 shows the thread block size selection process guided by occupancy in the LLVM compilation framework.

At first, SCoPs is extracted from the LLVM-IR of source program, and then PLuTo algorithm is used to obtain the schedule tree. After determining the nodes that can be mapped to different platforms, mark nodes such as threads, thread blocks, thread synchronization and data copies are inserted into the tree. For the sub-tree that can be mapped to GPU, the LLVM-IR with thread identification is generated. For other parts of the schedule tree, the generated LLVM-IR is inserted into startup, data copy, and other GPU runtime calls.

The LLVM-IR of GPU platform will be translated into machine code or virtual machine code under the backend support provided by LLVM compilation framework, and embedded into the host code as a global symbol. In this process, machine code can be used to get the actual number of registers used and the shared memory utilization of the multiprocessor during kernel execution. Then, we use Eq. (1) to calculate the theoretical occupancy under different thread block sizes. The thread block size with the highest occupancy rate is selected, and the thread grid size is calculated based on the problem size of the kernel. After that, the configuration is completed in the host code. If there are multiple candidate thread block sizes, we put the block sizes in ascending order in order to ensure that the selected thread block size is appropriate. If the number of candidate sizes n is an even number, we take the $n/2$ th thread block size, otherwise we take the $\lfloor n/2 \rfloor$ th block size.

3.2 Improve the Warp Scheduling Space

For some kernels, there may be multiple thread block sizes that can achieve maximum occupancy. Generally speaking, there may be some correlation between multiple warps in the same thread block. If there are many instruction delay in kernels, the probability of instruction stalls tends to increase as the size of the thread block increases. In order to solve this problem, if there are complex array accesses or synchronization statements in kernels, we will directly select the smallest thread block size that meets the highest occupancy rate, but ensure that it is greater than 4 warp sizes.

When there are more array accesses in a kernel, there will be a lot of global memory access requirements during the kernel execution, which will easily cause the instruction pipeline stagnation. To assess the complexity of array access that exists in the kernel, we define two types of data reuse that can occur during kernel execution firstly:

Definition 1. (Data reuse within iterations) If \mathbf{i}, \mathbf{j} are vectors composed of statement instance variables executed in the same iteration of statement $S1$ and $S2$ respectively, we have.

$$\mathbf{i} \rightarrow \mathbf{j} : f_k^1(\mathbf{i}) = f_k^2(\mathbf{j}) \wedge \left\lfloor \frac{f_m^1(\mathbf{i})}{C} \right\rfloor = \left\lfloor \frac{f_m^2(\mathbf{j})}{C} \right\rfloor, k = 1, 2, \dots, m - 1 \quad (4)$$

Definition 2. (Data reuse between iterations) If \mathbf{i} , \mathbf{i}_p are the vectors composed of two adjacent statement instance variables under the specified schedule of statement S , we have.

$$\mathbf{i} \rightarrow \mathbf{i}_p : f_k(\mathbf{i}) = f_k(\mathbf{i}_p) \wedge \left\lfloor \frac{f_m(\mathbf{i})}{C} \right\rfloor = \left\lfloor \frac{f_m(\mathbf{i}_p)}{C} \right\rfloor, k = 1, 2, \dots, m - 1 \quad (5)$$

Where C is the access granularity of global memory. Data reuse within an iteration describes whether array accesses generated by multiple polyhedral statements within the same iteration exist within the same global memory access granularity. The array accesses with data reuse within iterations in the thread can be combined to access to reduce the number of memory accesses when these statements are mapped to the GPU [18].

When there are still loops in the generated kernel, data reuse between iterations describes whether two temporally adjacent array elements accessed by threads exist in the same global memory access granularity. At the same time, in a kernel without loops, data reuse between iterations describes whether two array elements accessed by two adjacent threads are in the same global memory access granularity.

Therefore, we use the number of independent memory access patterns that do not have two kinds of data reuses with other memory access to evaluate the complexity of array access in kernels. Algorithm 1 describes how to evaluate the array access complexity when given a schedule and a set of strongly connected components to be mapped to GPU.

For a set of strongly connected components to be mapped to the GPU, Algorithm 1 evaluates the reuse of data in iterations between statements within the set under certain schedule, and aggregates them into array access patterns. If there is a data reuse relationship within iteration between an array access and any array access in a certain access pattern, it is added to the pattern. When an array access does not have a data reuse relationship with any existing patterns, a new array access pattern is added for the array access. This algorithm performs such an analysis on each array, counts the number of independent array access patterns finally obtained and uses it as the array access complexity. In addition, for each independent array access pattern, if it does not have data reuse between iterations, the array access complexity is increased by one.

Algorithm 1 Compute array access complexity

Input: Strongly connected component set C , schedule S , and corresponding array access relationship list AccessMapList . Where AccessMapList is unique, that is, if there are multiple accesses to an array by a statement in the List , the access functions for those accesses must be different.

Output: Array access complexity of C under schedule S .

```

1:  $P_C^S = 0$ 
2: For each  $A_i$  in  $\text{AccessMapList}$ 
3:  $\text{mapList} \leftarrow$  search maps related to  $A_i$  in  $\text{AccessMapList}$ 
4:  $\text{reusePatternList} = []$ 
5: For each  $\text{map}$  in  $\text{mapList}$ :
6:   For each  $\text{pattern}$  in  $\text{reusePatternList}$ 
7:     For each  $\text{patternMap}$  in  $\text{pattern}$ 
8:       Extract schedules for  $\text{map}$  and  $\text{patternMap}$ 
9:     If satisfy Definition 1.
10:       $\text{pattern} \leftarrow \text{map}$ , goto 5. /*aggregating access into existing patterns*/
11:   End
12: End
13:  $\text{newPattern} = []$ 
14:  $\text{newPattern} \leftarrow \text{map}$  /*add a new access mode*/
15:  $\text{reusePatternList} \leftarrow \text{newPattern}$ 
16: End
17: For each  $\text{pattern}$  in  $\text{reusePatternList}$ 
18:   If not satisfy Definition 2.
19:      $P_C^S = P_C^S + 1$ 
20: End
21:  $P_C^S = P_C^S + \text{len}(\text{reusePatternList})$ 
22: End

```

Since most kernels will not have complex array access, we can set a threshold for this complexity. If the number of independent array access patterns in the kernel exceeds this threshold, the smallest thread block size will be selected from the block sizes with the highest occupancy, but it should be ensured that it is greater than or equal to 128, which is the size of 4 warps. On the device used in this paper, we set the threshold to 5.

Besides, since there may be some correlations between multiple warps belonging to the same thread block, the more warps in a thread block, the greater the synchronization overhead. When there are many synchronizations and control flow in kernel program, a smaller thread block size should be used so that the instruction delay can be hidden as much as possible between warps belonging to different thread blocks. In the scenario of using polyhedral model to automatically generate GPU code, due to the need to ensure the correctness of the program transformation, there are often many synchronization statements in the generated kernel. Therefore, when the kernel has

control flow or synchronization statements inside loops, our strategy also selects the smallest thread block size that meets the occupancy requirements.

4 Evaluation

4.1 Experimental Environment

We implemented this thread block selection strategy in Polly-Acc of the LLVM compilation framework. The relevant tool chain version is Clang/LLVM10, isl-0.20, PolyBench 4.2.1, and CUDA 10.2. We tested this strategy on NVIDIA Tesla P100, which has a memory capacity of 16281MB, 56 multiprocessors, and 3584 stream processors. The computing capacity of this GPU is 6.0.

The comparison test included the use of three fixed block sizes in Polly-Acc: 32×16 , 32×8 , 32×4 and the thread block size calculated by *cudaOccupancyMaxPotentialBlockSize* in the CUDA API. Some test cases in PolyBench that are sensitive to the block sizes are selected as test cases, which involve many fields, such as linear algebra, image processing and dynamic programming, etc. In order to ensure the proper size of kernels generated during the automatic code generation and minimize the number of kernels, the test adopts the incremental scheduling method in the isl library to make decisions about loop fusion.

4.2 Results Analysis

The polyhedral compilation module performs affine transformations to explore parallelism without destroying dependencies. In the process of GPU thread mapping, the parallel loop is automatically tiled according to the thread block and thread grid size, and thread marking and memory management are performed. This ensures the correctness of the program for automatic code generation using different thread block sizes.

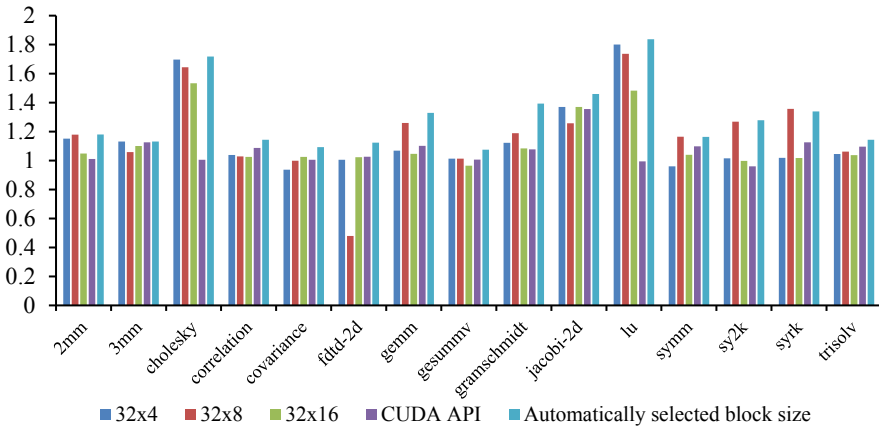


Fig. 2. Performance comparison with different thread block sizes relative to 32x32.

Figure 2 shows the test results under multiple thread block size strategies. In general, compared to using the 32x32 thread block size for all kernels, the thread block selection strategy used in this article enables the GPU automatic code generation to obtain an average performance improvement of 29% on this test set, and achieved 15.5% performance improvement compared to the default block size. On the other hand, *cudaOccupancyMaxPotentialBlockSize* API selects the maximum block size for most kernels. In all the test cases except Correlation, Trisolv and Gemm, the optimal performance was not achieved. Furthermore, this method achieves the worst performance in both Lu and Cholesky test cases. Compared to using this API, our approach achieved a performance improvement of approximately 21%. For most test cases, the selection strategy selects 256 for block size based on the occupancy, which is consistent with the default block size. For programs such as Jacobi-2d, Symm and Syrk, their kernels does not have complex synchronization statements or array accesses. Take the Jacobi-2d program as an example, on a device with compute capacity of 6.0, two kernels have same occupancy curve, and the corresponding performance is shown in Fig. 3. For these kernels, choosing a high-occupancy and medium-sized block size will often achieve better performance.

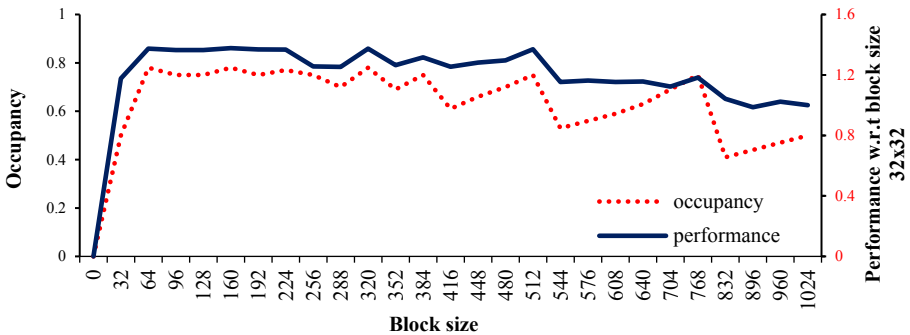


Fig. 3. The occupancy of Jacobi-2d kernels and related performance.

In order to increase the warp scheduling space, small block sizes are selected in some kernels in programs such as Lu and Cholesky. Figure 4 shows a kernel generated for Lu, which contains complex control flow and synchronization statements in nested loops. This will cause frequent pipeline stalls in threads during execution.

For a kernel with many control flows or synchronization statements, we can evaluate the actual performance by calculating the theoretical occupancy when the block size is within smaller range. However, as the size of the thread block increases, the warp scheduling space on the multiprocessor decreases, resulting in more pipeline stops during the execution of threads and poor actual performance. We can clearly observe in Fig. 5 that with the increase of block size, although the occupancy can still reach the highest value, but the actual performance has a significant decline. By using our block size selection strategy, a block size of 128 is used for the two kernels with complex synchronization statements and control flows in Lu, and a block size of 256 is

```

__global__ void kernel2(double *A, int n, int c0)
{
    int b0 = blockIdx.y, b1 = blockIdx.x;
    int t0 = threadIdx.y, t1 = threadIdx.x;
    __shared__ double shared_A_1[32][1];
    __shared__ double shared_A_2[1][32];

    for (int c1 = 32 * b0 + 8192 * ((-32 * b0 + c0 + 8161) / 8192);
        c1 < 32 * (b0 - b1 + 255) % 256 + n - 8160; c1 += 8192) {
        if (t1 == 0 && n >= t0 + c1 + 1)
            shared_A_1[t0][0] = A[(t0 + c1) * n + c0];
        __syncthreads();
        for (int c2 = 32 * b1 + 8192 * ((-32 * b1 + c1 + 8160) / 8192);
            c2 < n; c2 += 8192) {
            if (t0 == 0)
                for (int c4 = t1; c4 <= min(31, n - c2 - 1); c4 += 8)
                    shared_A_2[0][c4] = A[c0 * n + (c2 + c4)];
                __syncthreads();
            if (t0 + c1 >= c0 + 1)
                for (int c4 = max(t1, t1 + 8 * fdiv_q(t0 - t1 + c1 - c2 - 1, 8) + 8);
                    c4 <= min(31, n - c2 - 1); c4 += 8)
                    A[(t0 + c1) * n + (c2 + c4)] -= (shared_A_1[t0][0] *
                    shared_A_2[0][c4]);
                __syncthreads();
            }
        }
    }
}

```

Fig. 4. Source code corresponding to a kernel with synchronization statements in Lu.

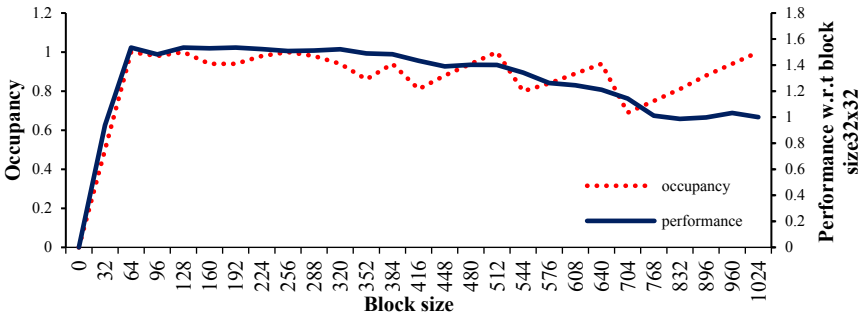


Fig. 5. The occupancy of kernel2 in Lu and related performance.

used for the remaining kernels. Compared with using a fixed 256 thread block size, the performance is improved by about 6%.

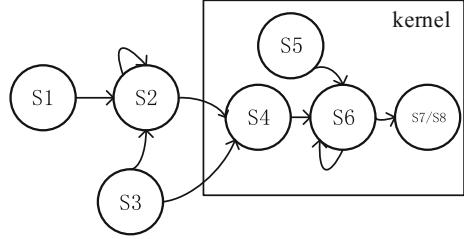
For the Covariance program fragment and its corresponding schedule graph as shown in Fig. 6(a) and 6(b). After the polyhedral scheduling is completed, S4-S8 are fused in the same outer loop and mapped to the GPU.

```

for (j = 0; j < _PB_M; j++)
{
S1  mean[j] = 0.0;
    for (i = 0; i < _PB_N; i++)
S2  mean[j] += data[i][j];
S3  mean[j] /= float_n;
}
for (i = 0; i < _PB_N; i++)
for (j = 0; j < _PB_M; j++)
S4  data[i][j] -= mean[j];
for (i = 0; i < _PB_M; i++)
for (j = i; j < _PB_M; j++)
{
S5  cov[i][j] = 0.0;
    for (k = 0; k < _PB_N; k++)
S6  cov[i][j] += data[k][i] * data[k][j];
S7  cov[i][j] /= (float_n - 1.0);
S8  cov[j][i] = cov[i][j];
}

```

(a)



(b)

Fig. 6. The Covariance program fragment and its corresponding schedule graph.

For the connected component set composed of strongly connected components S4-S8, the number of independent array access patterns is 6, and for the access of data $[k][j]$, $data[k][i]$ in the innermost loop, there is no data reuse between iterations. So the array access complexity of the strongly connected component set is 8. In this case, it will also cause a lot of pipeline stalls during thread execution, so a smaller thread block size should be used.

In a common application, multiple kernels may be generated in each SCoP. When the thread block size is fixed for each kernel, some kernels will get poor performance. This phenomenon occurs in programs such as Correlation, Gramschmidt, and Trisolv, which have kernels with different registers and shared memory usage. Using the Correlation program as an example, Table 1 shows the characteristics of 8 kernels in Correlation.

Table 1. Characteristics of eight kernels in Correlation

Kernel	Resource utilization	L/S complexity	Synchronous inside loops	CUDA API	Our method
0	13 reg	1	0	32×32	32×8
1	19 reg	1	0	32×32	32×8
2	13 reg	1	0	32×32	32×8
3	32 reg	4	0	32×32	32×8
4	31 reg	3	1	32×32	32×6
5	46 reg, 512B smem	1	2	32×21	32×4
6	40 reg, 8192B smem	5	2	32×25	32×6
7	25 reg	1	0	32×32	32×8

In actual performance tests, kernel4, kernel5, and kernel6 have higher time consumption than other kernels due to the intensive calculation and memory access. The theoretical occupancy of these kernels under different thread block sizes are shown in Fig. 7.

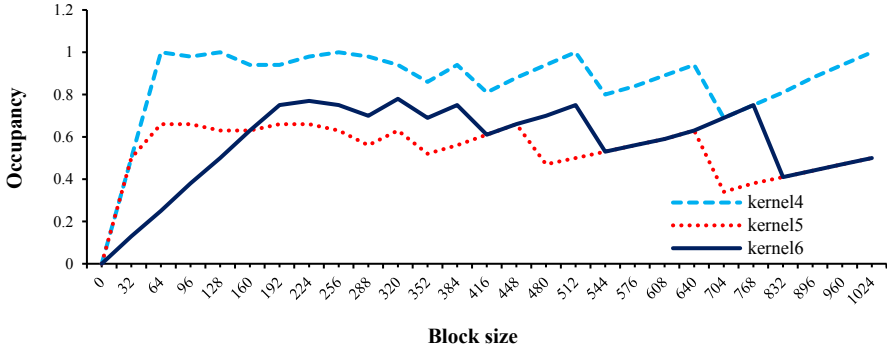


Fig. 7. The occupancy curve of the three kernels in Correlation under different thread block sizes.

In the test of using CUDA API for block size selection, kernel4, Kernel5 and Kernel6 obtained the largest block size that conforms to the highest occupancy rate, which was 1024, 800 and 672 respectively. Although it achieved better performance compared to other tests of fixed block size, it still has not achieved optimal performance. In the method proposed in this paper, for kernel 6, which has two synchronization statements within loop nests, we directly select the block size of 192 with an occupancy of 78%. Similarly, for kernel4 and kernel5, block sizes of 192 and 128 are selected respectively. For the remaining kernels with low access complexity and fewer synchronization statements, the thread block size 256 is selected. Experimental results show that compared with using CUDA API guided block size, this strategy improves the test performance of Correlation by 5%.

5 Conclusions and Future Work

In the scenario of automatic generation of GPU code based on the polyhedral model, our strategy can automatically select a better thread block size by ensuring the occupancy of kernels and increasing the warp schedule space. This strategy supports more flexible thread block size configurations and performance improvements than using user-specified thread block sizes. The results on PolyBench demonstrate the effectiveness of this method. However, although this strategy guarantees the occupancy and warp schedule space, the performance bottleneck of the GPU kernel lies in memory accesses in many cases. In order to find more suitable thread block size for different kernels, how to apply the analysis capabilities of the polyhedral model to the memory access of GPU platforms still needs further research.

References

1. Chen, T., Moreau, T., et al.: TVM: an automated end-to-end optimizing compiler for deep learning. In: OSDI'18: Proceedings of the 13th USENIX conference on Operating Systems Design and Implementation, pp. 579–594. ACM Press, New York (2018)
2. Vasilache, N., Zinenko, O., Theodoridis, T., et al.: Tensor comprehensions: framework-agnostic high-performance machine learning abstractions (2018)
3. Baghdadi, R., Ray, J., Romdhane, M.B., et al.: Tiramisu: a polyhedral compiler for expressing fast and portable code (2018)
4. Baskaran, M.M., Ramanujam, J., Sadayappan, P.: Automatic C-to-CUDA code generation for affine programs. In: Gupta, R. (ed.) *Compiler Construction. Lecture Notes in Computer Science*, vol. 6011, pp. 244–263. Springer, Berlin (2010). https://doi.org/10.1007/978-3-642-11970-5_14
5. Rudy, G.: *CUDA-CHiLL: a programming language interface for GPGPU optimizations and code generation. Dissertations & Theses – Gradworks* (2010)
6. Verdoolaege, S., Juega, J., Cohen, A., Gomez, J.I., Tenllado, C., Cattloor, F.: Polyhedral parallel code generation for CUDA. *ACM Trans. Archit. Code Optim. (TACO)* **9** (2013). Article no. 54
7. Zhao, J., Li, Y.Y., Zhao, R.C.: “Black magic” of polyhedral compilation. *J. Softw.* **29**(8), 2371–2396 (2018). (in Chinese)
8. Bondhugula, U., Baskaran, M., Krishnamoorthy, S., Ramanujam, J., Rountev, A., Sadayappan, P.: Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model. In: Hendren, L. (ed.) *CC 2008. LNCS*, vol. 4959, pp. 132–146. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78791-4_9
9. Leung, A., Vasilache, N., Meister, B., et al.: A mapping path for multi-GPGPU accelerated computers from a portable high level programming abstraction, p. 51 (2010)
10. Pouchet, L.N., Grlinger, A., Simbürger, A., et al.: Polly-polyhedral optimization in LLVM. In: *International Workshop on Polyhedral Compilation Techniques (IMPACT)* (2011)
11. Grosser, T., Hoefler, T.: Polly-ACC transparent compilation to heterogeneous hardware. In: *International Conference on Supercomputing. ACM* (2016)
12. Amini, M., Coelho, F., Irigoien, F., Keryell, R.: Static compilation analysis for host-accelerator communication optimization. In: Rajopadhye, S., Mills Strout, M. (eds.) *LCPC 2011. LNCS*, vol. 7146, pp. 237–251. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36036-7_16
13. Shobaki, G., Kerbow, A., Mekhanoshin, S.: Optimizing occupancy and ILP on the GPU using a combinatorial approach. In: *Proceedings of the 18th ACM/IEEE International Symposium on Code Generation and Optimization (CGO 2020)*, pp. 133–144. Association for Computing Machinery, New York (2020)
14. Nickolls, J.: Scalable parallel programming with CUDA introduction. In: *2008 IEEE Hot Chips 20 Symposium (HCS)*, Stanford, CA, pp. 1–9 (2008). <https://doi.org/10.1109/HOTCHIPS.2008.7476518>
15. Feautrier, P.: Some efficient solutions to the affine scheduling problem. Part II . Multidimensional time. *Int. J. Parallel Prog.* **21**(6), 389–420 (1997)
16. Grosser, T., Verdoolaege, S., Cohen, A.: Polyhedral AST generation is more than scanning polyhedra. *ACM Trans. Program. Lang. Syst.* **37**(4), 12 (2015)

17. Hayes, A., Li, L., Chavarría-Miranda, D., Song, S., Zhang, E.: Orion: A Framework for GPU Occupancy Tuning, pp. 1–13 (2016). <https://doi.org/10.1145/2988336.2988355>
18. Fauzia, N., Pouchet, L.-N., Sadayappan, P.: Characterizing and enhancing global memory data coalescing on GPUs. In: Proceedings of the 13th Annual IEEE/ACM International Symposium on Code Generation and Optimization (CGO 2015), pp. 12–22. IEEE Computer Society, USA (2015)



SPORTS: A Semi-partitioned Real-Time Scheduler for Heterogeneous Multicore Platforms

Yanshul Sharma, Zinea Das, and Sanjay Moulik^(✉)

Indian Institute of Information Technology Guwahati (IIITG), Guwahati, India
sanjay@iiitg.ac.in

Abstract. Over the years, the nature of processing platforms is witnessing a shift from homogeneous to heterogeneous multicore, where several applications share the same platform. Also, these systems have to perform complex functionalities on densely packed multi-million gate platforms, which has made resource usage efficiency more challenging to handle. In this work, we propose a two-phase hierarchical resource allocation strategy called *SPORTS: A semi-partitioned real-time scheduler for heterogeneous multicore platforms*, for scheduling of periodic tasks with bounded number of migrations and context-switches. In the first phase, it divides time into distinct intervals/windows based on deadlines of tasks so that exact proportional fairness needed for progress of task executions is maintained at all window boundaries. In the second phase, it applies an efficient heuristic to perform task-to-core assignments based on task's execution requirements. Our experimental analysis shows that the presented strategy is able to significantly improve (up to 11.64% on an average) upon state-of-the-art in terms of resource utilisation.

Keywords: Heterogeneous · Multicore · Real-time · Proportional fair · Resource-usage

1 Introduction

A system is classified as real-time if it is characterised by a dual notion of correctness: logical as well as temporal. In spite of progressively increasing computing capabilities of hardware platforms, effective allocation of computing resources to diverse competing applications is set to remain a daunting problem in real-time embedded systems. This is because, these systems impose stringent timing, resource and performance related constraints, which must be accurately captured in the design process in order to ensure proper functioning of the system. In this context, scheduling acts as a vital design component which determines an appropriate co-execution order for the tasks such that the desired performance objectives may be achieved while satisfying all constraints. Over the years, the nature of execution platforms is witnessing a shift from homogeneous to heterogeneous multicore, where several applications share the same platform. On a

heterogeneous platform, the same lines of code may need dissimilar time durations to execute over non-identical cores, which makes scheduling on such a platform more challenging [12,15].

Traditional schedulers for periodic tasks on multicore real-time systems make use of either a *partitioned* or *global* approach. In a *fully partitioned* approach, every task is assigned to a single core and each task is allowed to execute on the allotted core only. This approach has the advantage of transforming the multicore scheduling problem into a set of uni-core scheduling ones. An Integer Linear Program (ILP) based method to partition tasks on heterogeneous platforms has been addressed in [2]. However, solving ILPs for systems with a large number of tasks and cores may incur prohibitively expensive computational overheads in many real-time systems. Although schedulers based on the partitioned approach are easier to implement but the process of *partitioning* is provably *NP-hard* [7] for multicore platforms. Unlike partitioning, a *global* approach allows the migration of a task onto any available core at every scheduling point. The seminal work for global scheduling of tasks on heterogeneous multicores with arbitrary number of cores was proposed in [1]. It involved a two step process. In the first step, the workload of each task is distributed over the available cores of the platform and then in the second step, a method to construct a template schedule is presented. In [4], the authors prove the incompleteness of the second step of the work [1] and have proposed another template scheduling method based on a global approach. Due to the relaxation of allowing tasks to migrate among cores, schedulers based on the global approach may lead to very high migration overheads, which is not a desired design feature for modern systems.

Recently, researchers have started focusing on a hybrid third approach called *semi-partitioned* [11,14]. These schedulers divide the timeline into intervals often called *windows* or *time-slices*. Tasks are executed in a partitioned fashion within every interval and they globally resynchronise at the interval boundary. They offer high resource utilisations with lower migration/context-switch overheads. However, the design of an efficient scheduler for a heterogeneous platform is a challenging problem and researchers have recently started giving adequate focus to this problem. An efficient task-to-core assignment strategy for heterogeneous platforms has been proposed in [16]. Subsequent to the assignment phase, the heterogeneous scheduling problem is transformed to homogeneous multicore scheduling. Then, they have applied well known optimal homogeneous multi-core scheduling techniques. Chwa et al. [8] extended an efficient scheduling strategy for homogeneous platforms named DPfair [9] to heterogeneous platforms. However, these approaches [8,16] have focused towards heterogeneous platforms with only two-types of cores (like ARM's big.LITTLE) and are not applicable to generic heterogeneous platforms consisting of more than two core types. In [13], a cluster based scheduling strategy based on a same semi-partitioned has been proposed, which form clusters of cores based on execution requirements of tasks before performing task-to-core assignments. However, they don't allow inter cluster migrations and hence, provide lower resource utilisations. Efficient usage of resources in devices like mobiles, laptops, PDAs, etc., while satisfying

all temporal constraints, has become a design parameter of paramount importance. Therefore, the proposed work primarily focuses on the problem of efficient resource usage, which has not been focussed adequately in the literature. Also, most of the works in literature, only considers the execution demands of tasks while performing task-to-core assignments and does not consider the execution variation of tasks on cores unlike our strategy, which may lead to lower number of scheduled tasks if the variation in execution requirements is high. Hence, our heuristic will fetch better results than existing works in such scenarios, as shown in the experiment results. The contributions of our work can be summarised as follows:

- i.** Development of an efficient task allocation strategy, which efficiently allocates tasks to a given heterogeneous platform using a two-phased mechanism with a bound on context-switch/migration related overheads.
- ii.** The designed scheduling strategy maintains relative fairness among the execution progress of tasks at all window boundaries.
- iii.** Analysis conducted using extensive simulation based experiments show that our proposed scheduling scheme is able to significantly improve acceptance ratios of task sets when the number of heterogeneous processing cores in the system increases, compared to state-of-the-art [13].

In our opinion, it fits more precisely to certain realistic platforms where cores have different micro-architectures but identical ISA, as the big.LITTLE[®] or the Helio X20[®].

2 Specifications

We considered a periodic task set $\tau = \{\tau^1, \tau^2, \dots\}$, which has to be scheduled on a heterogeneous multicore platform $\Pi = \{\Pi^1, \Pi^2, \dots\}$. Each occurrence of a periodic task τ^i is associated with a $(|\Pi| + 2)$ tuple, $\langle e^i, p^i, r^{i1}, r^{i2}, \dots, r^{i|\Pi|} \rangle$ where e^i and p^i are the execution requirement and the period (as well as deadline) of τ^i in the system and r^{ij} is the rate of execution of τ^i on Π^j . Hence, each task τ^i can further be related to the terms: $u^i (= e^i/p^i)$, which represents utilisation of τ^i in the system and, $u^{ij} (= e^i/(p^i \times r^{ij}))$, which represents utilisation of τ^i on a core Π^j . Each context-switch within the schedule (caused due to a task preemption or migration) is associated with a time overhead during which no useful task execution can happen on the core. In the current work, this overhead has been modeled by assuming the execution time of each task to be proportionately inflated. It may be noted that by following the approach of task execution time inflation to take care of context-switch overheads, we have been able to avoid the nuisances of explicitly accounting for context-switch overheads as part of the schedule generation process.

3 Problem Description

Given a set τ of implicit-deadline periodic tasks and a set Π of heterogeneous processing cores, the objective is to construct a feasible schedule, such that all task instances always meet their individual execution and deadline requirements.

Formal Problem Formulation as a CSP: We now formally model the problem discussed above as a *Constraints Satisfaction Problem (CSP)*. In the interest of generating a well-structured model which is easy to understand and analyse, we slightly restrict the problem at hand by assuming the set of tasks to be persistent and continuously running. This restriction allows the system to have complete knowledge of future workloads as dynamic task arrivals and departures are avoided. Hence, the overall schedule can be represented by the schedule for one hyper-period $H = lcm(p^1, p^2, \dots, p^{|\tau|})^*$, of the set of tasks. For a heterogeneous multicore system (τ, Π) , let x_t^{ij} represent a set of binary decision variables where:

$$x_t^{ij} = \begin{cases} 1, & \text{if } \tau^i \text{ executes on core } \Pi^j \text{ at time instant } t \\ 0, & \text{otherwise} \end{cases}$$

The set of feasible values that the decision variables can take over the length of the hyper-period H , are subject to the following set of constraints:

i. **Core Capacity Constraints:** At any time instant t , each core can execute at most one task.

$$\forall j, t (1 \leq j \leq |\Pi|; 1 \leq t \leq H) \sum_{i=1}^{|\tau|} x_t^{ij} \leq 1 \tag{1}$$

ii. **Task Execution Constraints:** No task can execute simultaneously on more than one core.

$$\forall i, t (1 \leq i \leq |\tau|; 1 \leq t \leq H) \sum_{j=1}^{|\Pi|} x_t^{ij} \leq 1 \tag{2}$$

iii. **Deadline Constraints:** All instances of each task must complete their execution requirements on or before their respective deadline. It may be noted that a task τ^i has H/p^i instances, with the s^{th} instance having a start time of $(s - 1) \times p^i$, to be completed within the period/deadline, $s \times p^i$.

$$\forall i, s (1 \leq i \leq |\tau|; 1 \leq s \leq H/p^i) \sum_{t=(s-1) \times p^i}^{s \times p^i} \sum_{j=1}^{|\Pi|} x_t^{ij} / r^{ij} = e^i \tag{3}$$

4 The Proposed Algorithm

SPORTS (Algorithm 1): The execution progresses in the system window by window. In order to do so, it starts by computing the hyper-period H using periods of the tasks (Line 3). Next, it computes windows by using the technique of *Deadline Partitioning* [9] (Line 4), in which the windows are demarcated by the periods/deadlines of all tasks in the system. Within a window W_k , the algorithm computes the execution demands/shares for each task belonging to the

*least common multiple.

task set τ on each core II^j for the ensuing window (Line 9) using the following equation:

$$sh^{ij} = \left\lceil \frac{e^i}{p^i \times r^{ij}} \times |W^k| \right\rceil \quad (4)$$

If a task is able to complete its execution requirement of $\frac{e^i}{p^i} \times |W^k|$ (assuming rate to be 1.0) within every window, then it is guaranteed to meet its total requirement within the deadline/period. This ensures deadline constraints discussed in Sect. 3. The specifications of each task τ^i is stored in a different list L_i (Line 10), where each node is of the form $\langle i, j, r^{ij}, sh^{ij} \rangle$. The list remains sorted in non-decreasing order of their share values. Each formation of each list L_i , it is added to the global task list LT_1 (Line 11). Next, it calls functions ALLOT-NM-TASKS() (Line 12) and ALLOT-M-TASKS() (Line 13), to perform task-to-core allocation.

Algorithm 1: SPORTS

Input: τ, II

Output: A set of schedule tables

- 1 Let i. $W = \{W_1, W_2, \dots\}$ be set of windows, ii. sc^j be II^j 's spare capacity, iii. L_i a be non-decreasing order sorted list based on shares of τ^i , iv. LT_1 be list of L_i 's and, v. LT_M be the migrating task list
 - 2 Initialise $k \leftarrow 0$; $time \leftarrow 0$
 - 3 Compute hyperperiod $H = lcm(p^1, p^2, \dots, p^{|\tau|})$
 - 4 Find windows in $[0, H]$ using *Deadline Partitioning*
 - 5 **for each window** $W_k \in W$ **do**
 - 6 Let sh and SM_k be share and schedule matrices
 - 7 **for** $i \leftarrow 1 : |\tau|$ **do**
 - 8 **for** $j \leftarrow 1 : |II|$ **do**
 - 9 Compute share requirement sh^{ij} (Eqn. 4)
 - 10 $L_i \leftarrow L_i \cup \{\langle i, j, r^{ij}, sh^{ij} \rangle\}$
 - 11 $LT_1 \leftarrow LT_1 \cup \{L_i\}$
 - 12 ALLOT-NM-TASK (LT_1, LT_M, SM_k, II)
 - 13 ALLOT-M-TASK (τ, LT_M, SM_k, II)
-

ALLOT-NM-TASKS (Algorithm 2): It performs task-to-core assignments for the tasks which can be fully allocated on a single core. These tasks are called non-migrating tasks. The function iterates over the LT_1 . At start of each iteration, it calls the function GET-ED-MAX() (Line 2) to get $Node_{max}$ (having format $\langle i, j, sh^{ij}, ED_i \rangle$), node of the task with maximum difference of execution requirements on its two most favored cores. If none of the unallocated tasks can be fully allocated on a single core, the function GET-ED-MAX() returns ϕ . In such a scenario, all remaining tasks are removed from LT_1 and added to the list of migrating tasks LT_M (Line 4). Otherwise, it updates the *Schedule Matrix* (SM_k) with the start and end time for τ^i on II^j (Line 6), updates the remaining

Algorithm 2: ALLOT-NM-TASKS

Input: LT_1, LT_M, SM_k, Π **Output:** SM_k and LT_M

```

1 while  $LT_1 \neq \phi$  do
2    $Node_{max} \leftarrow$  GET-ED-MAX ( $LT_1, \Pi$ )
3   if  $Node_{max} = \phi$  then
4     Remove all  $L_i$ 's from  $LT_1$  and set  $LT_M \leftarrow LT_M \cup \{L_i\}$ 's
5   else
6     Assign start and end times of  $\tau^i$  on  $\Pi^j$ , i.e.,
7      $SM_k[i][j] \leftarrow \langle \text{start\_time}(\tau^i), \text{end\_time}(\tau^i) \rangle$ 
7     Set  $sc^j \leftarrow sc^j - sh^{ij}$  and remove  $L_i$  from  $LT_1$ 

```

spare capacity sc^j of Π^j and removes L_i from LT_1 (Line 7). It may be noted that a task can only start its execution on a core after the task which has been previously allocated on Π^j finishes.

Algorithm 3: GET-ED-MAX

Input: LT_1, Π **Output:** $Node_{max}$

```

1 Let  $Node_{max}$  be the node corresponding to the task having maximum difference
  in shares on its two most favored cores
2 for each  $L_i \in LT_1$  do
3   Extract first two nodes from  $L_i$ , where  $sc^j > sh^{ij}$  (say on cores  $j_1$  and  $j_2$ )
4   if only one such core  $\Pi^{j_1}$  exists then  $ED_i \leftarrow sh^{ij_1}$ 
5   else  $ED_i \leftarrow sh^{ij_2} - sh^{ij_1}$ 
6   if  $ED_i > Node_{max}.ED_i$  then  $Node_{max} \leftarrow \langle i, j_1, sh^{ij_1}, ED_i \rangle$ 

```

GET-ED-MAX (Algorithm 3): It returns the node $Node_{max}$ corresponding to the task having maximum difference in execution requirements on its two most favored cores. It iterates over LT_1 , which has been given to it as an input. In each iteration, it considers one L_i . It extracts the minimum execution requirements for τ^i on two cores (say Π^{j_1} and Π^{j_2}) which have spare capacities greater than requirements on them (Line 3), which ensures the core capacity constraints discussed in Sect. 3. Then, it calculates the variation $ED_i = sh^{ij_2} - sh^{ij_1}$ (Line 5). This value signifies the penalty which has to be paid if τ^i is not allocated to Π^{j_1} . If only one such core Π^{j_1} exists then set $ED_i = sh^{ij_1}$ (Line 4). If the computed ED_i value is greater than the stored ED_i value of $Node_{max}$ then the values in $Node_{max}$ are updated accordingly (Line 6). *Most of the works in literature, only considers the minimum execution demands of tasks while performing task-to-core assignments, which may lead to lower number of scheduled tasks if the variation in execution requirements of tasks on cores is high.* Hence, our heuristic

will fetch better results than existing works in such scenarios, as shown in the experiment results.

Algorithm 4: ALLOT-M-TASKS

Input: τ , LT_M , SM_k , Π
Output: SM_k

```

1 while  $LT_M \neq \phi$  do
2   Get node  $L_i$  from  $LT_M$ 
3   Let  $us^i$  be the unallocated share of  $\tau^i$ 
4   while  $L_i \neq \phi$  do
5     Get the first entry  $\langle i, j, r^{ij}, sh^{ij} \rangle$  from  $L_i$ 
6     if  $us^i / r^{ij} > sc^j$  then
7       Update  $SM_k[i][j]$  to schedule  $\tau^i$  on  $\Pi^j$  for duration  $sc^j$ 
8        $us^i \leftarrow us^i - sc^j \times r^{ij}$  and  $sc^j \leftarrow 0$ 
9     else
10      Update  $SM_k[i][j]$  to schedule  $\tau^i$  on  $\Pi^j$  for duration  $us^i / r^{ij}$ 
11      Update  $sc^j \leftarrow sc^j - (us^i / r^{ij})$  and  $us^i \leftarrow 0$ 
12      Remove all entries of  $L_i$ 
13   if  $us^i \neq 0$  or  $\tau^i$  is allocated on multiple cores in parallel then
14     Reject  $\tau^i$  from system

```

ALLOT-M-TASKS (Algorithm 4): It iterates over all elements of the list LT_M . In each iteration, it extracts a node L_i from the list LT_M (Line 2). Then, it considers all elements of L_i sequentially. Let, the *unallocated share* of τ^i be denoted as us^i and $\langle i, j, r^{ij}, sh^{ij} \rangle$ be the element under consideration. Note, initially $us^i = (e^i \times |W_k|) / p^i$, where the rate of execution has been considered as 1.0. In our experiments (discussed later), we have set the rate value of 1.0 on at least one core for each task. There can be following two cases which may arise while allocating τ^i on Π^j :

- $us^i / r^{ij} > sc^j$: In such a case, we partially allocate τ^i on Π^j for duration sc^j (Line 7) and update us^i (Line 8).
- $us^i / r^{ij} \leq sc^j$: In this case, we fully allocate τ^i on Π^j for duration us^i / r^{ij} (Line 10) and the spare capacity of Π^j is updated (Line 11). Then, all nodes for τ^i are removed from L_i (Line 12).

The solutions for the above written cases ensure the core capacity constraints discussed in Sect. 3. In order to schedule a migrating task τ^i , SPORTS uses the following heuristic to avoid parallel execution of the same task on multiple cores simultaneously and meet task execution constraints discussed in Sect. 3. When a migrating task is allocated partially on the first core, it is scheduled from the beginning of the window on that core. From the next core onwards, the migrating task is going to start its execution on a core after it finishes its

execution on the core where it was last partially allocated. The starting and ending times of execution for each task on the cores are stored in SM_k . If there is not sufficient capacity in the system to schedule a task τ^i , then the task is not considered for scheduling in the system any further.

5 Analysis of the Algorithm

Analysis of Time-Complexity: Let us analyse each of the function used in the algorithm one by one:

Function GET-ED-MAX(): There are $|\tau|$ nodes in LT_1 . Each node L_i in LT_1 is sorted, hence, ED_i computation takes $O(1)$ time. Therefore, time complexity of the function GET-ED-MAX() is $O(|\tau|)$.

Function ALLOT-NM-TASKS(): It iterates till LT_1 is empty, hence $|\tau|$ times. In each iteration, it either gets the task with minimum difference in execution requirement in $O(|\tau|)$ time or adds all nodes of LT_1 to LT_M , which takes $O(|\tau|)$ time. Hence, the overall time complexity of the function becomes $O(|\tau|^2)$.

Function ALLOT-M-TASKS(): For each migrating task node L_i in LT_M , it may check for its allocation on every core of the system, which will require $O(|II|)$ time. There can be $|\tau|$ nodes in LT_M . Hence, the time complexity of the function becomes $O(|\tau| \times |II|^2)$.

Function SPORTS(): The computation of the hyper-period is done only once, so it can be neglected. Finding size of the next window takes $O(|\tau|)$ time. For each task, it computes share on all cores, which takes $O(|II|)$ time. Further, it constructs a sorted list L_i for each task, which again takes $O(|II|^2)$ time. Then, it calls functions ALLOT-NM-TASKS() and ALLOT-M-TASKS(). So, the time-complexity of the function SPORTS() becomes $O(|\tau|^2 \times |II|^2)$ ($=O(|\tau|^2 \times |II|^2) + O(|\tau| \times |II|)$). Assuming the number of cores $|II|$ in a platform to be much less than the number of tasks $|\tau|$, the overall time complexity of the algorithm becomes $O(|\tau|^2)$ per window.

Migration and Context-Switch Bound: In our approach, when a task cannot be fully allocated to any single core, it is carefully allocated and scheduled across multiple cores in a window such that its execution on a core does not overlap with any other core in the system. From the partitioning and scheduling mechanism discussed above, we may observe that a particular core in the system may hold a set of non-migrating tasks along with either: i. no other migrating tasks, ii. multiple terminating migrating tasks, iii. one non-terminating migrating task, and iv. one non-terminating migrating task and multiple terminating migrating tasks. Cores having fixed tasks and terminating migrating tasks does not cause any migration in a window. Whenever a non-terminating migrating task is scheduled on a particular core, that core will not have any remaining

capacity to allot any other task. Hence, the number of migrations on a core with a non-terminating migrating task is restricted to only 1. There can be at most $|II| - 1$ cores with non-terminating migrating tasks in any window. Therefore, the number of migrations in a window using our strategy is bounded by $|II| - 1$.

At most $|\tau|$ tasks can be eligible for execution in a window, which will require $|\tau| - 1$ context-switches. As stated earlier, each core may have a non-terminating migrating task scheduled on it. Scheduling of each of such migrating task may lead to splitting of a non-migrating task on each core. Hence, there can be at most $|II|$ such splits, which bounds the total number of context-switches within each window to $|\tau| + |II| - 1$.

6 Experimental Set Up and Results

We have compared the performance of SPORTS against COST [13], a cluster oriented scheduler for heterogeneous multicore platforms with arbitrary number of cores, like SPORTS. It is based on the semi-partitioned approach and hence, offers efficient resource utilisation in the system.

Experimental Set Up: All our simulations have been run for a total execution time of 100000 time-slots with task sets having pre-specified *utilisation factors*. *Utilisation Factor (UF)* is defined as the ratio between the *summation over utilisations of tasks in the system* and the *number of available cores*. That is, $UF = \sum_{i=1}^{|\tau|} u^i / |II|$. For generating task sets with a particular *UF* value, the arbitrarily produced utilisation values have been scaled appropriately. Further, for each task τ^i , we have chosen a random core II^j on which its rate of execution is set to 1.0 (i.e. on II^j , $u^i = u^{ij}$) and for all other cores, its rate has been generated in the range 0.6 to 1.0. For each set of input parameters, we ran the simulation on 50 different test cases. The average of these 50 test cases has been considered as the final outcome. In order to compare the performance of SPORTS against COST, we have introduced two metrics namely, *Acceptance Ratio (ARat)*, and *Context-Switch Overhead (CSO)*. *ARat* measures the percentage of tasks that have been scheduled successfully by the algorithms, while *CSO* gives a measure of the context-switch overheads of the algorithms.

Table 1. Execution requirements of programs for Parsec [17] and Mälardalen benchmarks [10]

Application	Execution time (in ms)	Application	Execution time (in ms)	Application	Execution time (in ms)
body	3120	stream	11820	select	135
can	12300	swap	34500	qurt	130
fluid	960	x264	60	ndes	8340
freq	1440	bsort	9	lms	146
duff	73	edn	62		

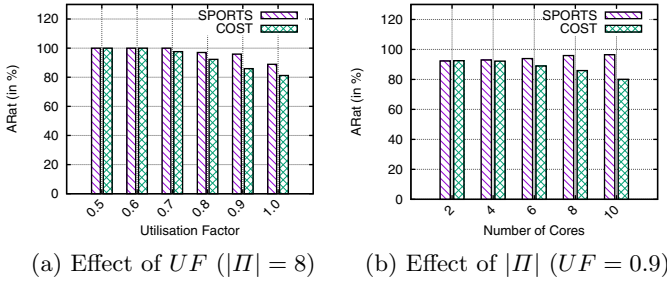


Fig. 1. Benchmark Program Results ($|\tau| = 14$) (Effect on ARat values)

Benchmarks Programs: We selected a set of 14 tasks from Parsec [17] and Mälardalen [10] benchmarks to evaluate the performance of the algorithms on systems with varying number of cores. A detailed discussion on the procedure for measuring execution requirements of each program is presented in [6] and the results have been listed in Table 1. These values have been obtained with the help of Gem5 [5] simulator for an ARMv8 processor (considering 32 nm CMOS technology), operating at 3.0 GHz. Each task set considered consists of 50 tasks, whose instances have been selected randomly from the benchmark applications (repeated to prepare the task set) listed in Table 1. To associate each task τ^i with the generated utilisation value u^i , its period has been appropriately generated.

Experiment 1: In this experiment, we observed the variation in ARat values with the increase in workload. We may observe from Fig. 1a, ARat values decrease with an increase in UF values. This phenomenon may be attributed to the fact that an increased workload results in a higher probability of tasks requiring migrations within windows. As both the algorithms are heuristic, they may fail to schedule all the tasks at higher workloads. In such a scenario, the task with the highest share in the window is rejected from further scheduling, resulting in reduced ARat values. COST forms cluster of cores based on execution requirements of tasks, with each cluster having at most 2 cores. It only

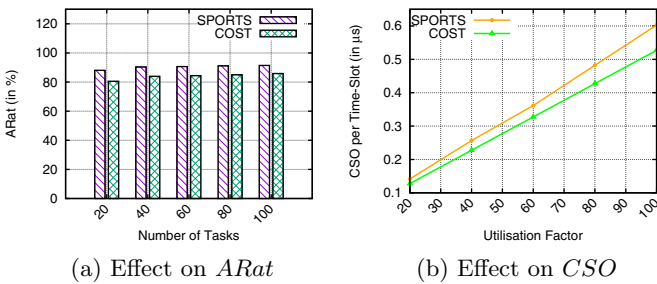


Fig. 2. Synthetic Task Set Results ($|\tau| = 8$ and $UF = 0.9$) (Effect of number of tasks)

allows migration within a cluster. Hence, with a better resource allocation heuristic, SPORTS is able to outperform COST progressively. From Fig. 1a, we may observe that ARat values decrease from 100% to 81.2% and 88.9% with the variation in UF values from 0.5 to 1.0, for COST and SPORTS, respectively.

Experiment 2: In this experiment, the number of cores have been varied from 2 to 10. As discussed before, COST only allows intra-cluster migration. Hence, an increase in the number of cores lead to higher requirement of inter-cluster migration and thus reduced ARat values. On the other hand, SPORTS allows unrestricted migrations within a system and an increase in the number of cores help the proposed algorithm to prepare the schedule in an easier way, thus leading to higher ARat values. We can observe from Fig. 1b, the ARat values vary from 92.5% to 80.1% and 92.3% to 96.5% with an increase in $|II|$ values from 2 to 10, for COST and SPORTS, respectively.

Synthetic Tasks: The experimentation framework used in this work considers a system with 8 cores. Further, it considers randomly generated task sets whose sizes ($|\tau|$) vary from 20 to 100. The task execution requirements have been generated from a normal distribution, having $\sigma^e = 100$ and $\mu^e = 20$. To test the efficiencies of the strategies, we have considered systems with high utilisation factor of 0.9. To associate each task τ^i with the generated utilisation factor u^i and execution requirement e^i , its period has been appropriately generated.

Experiment 3: In this experiment, the number of tasks have been varied from 20 to 100. We can observe from Fig. 2a that ARat values increase progressively for both strategies with an increase in the number of tasks. This phenomenon may be attributed to the fact that an increase in the number of tasks while having a constant number of cores and workload, leads to decrease in average execution requirements of individual tasks, which further results in decrease in the number of migrations for the algorithms. Hence, both the algorithms are able to have better ARat values with an increase in $|\tau|$ values. However, SPORTS shows better ARat values because of its principle of allowing full migration as opposed to only intra-cluster migration for the COST strategy. In particular, the ARat values increase from 80.55% to 85.85% and 88.05% to 91.4% with an increase in $|\tau|$ values from 20 to 100, for COST and SPORTS, respectively.

Experiment 4: We assumed that the timing delay for each context-switch corresponds to $5.24 \mu s$ [3], which represents the average value of a context-switch on a multi-core system under typical workloads. In order to find the total delay caused by context-switches in our experiment, we computed the number of context-switches for each run and then multiplied it with the delay due to a single context-switch ($5.24 \mu s$ [3]). Then, we computed the average overheads for context-switches per time-slot (in μs) for both the algorithms. As we can observe from Fig. 2b, the context-switch/migration overhead increases for both the algorithms with an increase in the number of tasks. However, the overhead for

SPORTS is slightly higher than COST, since COST does not allow inter-cluster migrations. However, as discussed above, use of such a strategy also leads to lower ARat values for COST, which is not desirable in a system.

7 Conclusion

In this work, we have presented a heuristic scheduling strategy named SPORTS, which provides an efficient resource usage on heterogeneous multicore platforms. The presented strategy considers execution requirements of given tasks on different cores, to perform scheduling in the system. Experimental results show that our scheme can deliver a significantly higher resource usage efficiency (up to 11.64% on an average) compared to state-of-the-art while incurring slightly higher context-switch/migration related overheads.

References

1. Baruah, S.: Feasibility analysis of preemptive real-time systems upon heterogeneous multiprocessor platforms. In: IEEE RTSS, pp. 37–46 (2004)
2. Baruah, S.K., Bonifaci, V., Bruni, R., Marchetti-Spaccamela, A.: ILP-based approaches to partitioning recurrent workloads upon heterogeneous multiprocessors. In: ECRTS, pp. 215–225 (2016)
3. Bastoni, A., Brandenburg, B.B., Anderson, J.H.: Cache-related preemption and migration delays: empirical approximation and impact on schedulability. In: OSPERT (2010)
4. Bertout, A., Goossens, J., Grolleau, E., Poczekajlo, X.: Template schedule construction for global real-time scheduling on unrelated multiprocessor platforms. In: DATE, pp. 216–221 (2020)
5. Binkert, N., et al.: The gem5 simulator. ACM SIGARCH **39**(2), 1–7 (2011)
6. Bygde, S., Ermedahl, A., Lisper, B.: An efficient algorithm for parametric WCET calculation. In: IEEE RTCSA, pp. 13–21, August 2009
7. Chattopadhyay, B., Baruah, S.: A lookup-table driven approach to partitioned scheduling. In: IEEE RTAS, pp. 257–265 (2011)
8. Chwa, H.S., Seo, J., Lee, J., Shin, I.: Optimal real-time scheduling on two-type heterogeneous multicore platforms. In: IEEE RTSS, pp. 119–129, December 2015
9. Funk, S., Levin, G., Sadowski, C., Pye, I., Brandt, S.: DP-Fair: a unifying theory for optimal hard real-time multiprocessor scheduling. *Real-Time Syst.* **47**(5), 389–429 (2011). <https://doi.org/10.1007/s11241-011-9130-0>
10. Gustafsson, J., Betts, A., Ermedahl, A., Lisper, B.: The Mälardalen WCET benchmarks - past, present and future. In: Lisper, B. (ed.) WCET, pp. 137–147. OCG, Brussels, Belgium, July 2010
11. Moulík, S., Sarkar, A., Kapoor, H.K.: DPFair scheduling with slowdown and suspension. In: 2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID), pp. 43–48 (2018). <https://doi.org/10.1109/VLSID.2018.35>
12. Moulík, S., Chaudhary, R., Das, Z.: HEARS: a heterogeneous energy-aware real-time scheduler. *Microprocess. Microsyst.* **72**, 102939 (2020). <https://doi.org/10.1016/j.micpro.2019.102939>, <http://www.sciencedirect.com/science/article/pii/S0141933119302017>

13. Moulik, S., Devaraj, R., Sarkar, A.: COST: a cluster-oriented scheduling technique for heterogeneous multi-cores. In: IEEE SMC, pp. 1951–1957. IEEE (2018)
14. Moulik, S., Sarkar, A., Kapoor, H.K.: Energy aware frame based fair scheduling. *Sustain. Comput. Inf. Syst.* **18**, 66–77 (2018). <https://doi.org/10.1016/j.suscom.2018.03.003>, <http://www.sciencedirect.com/science/article/pii/S2210537917300549>
15. Moulik, S., Sarkar, A., Kapoor, H.K.: Tarts: a temperature-aware real-time deadline-partitioned fair scheduler. *J. Syst. Archit.*, 101847 (2020). <https://doi.org/10.1016/j.sysarc.2020.101847>, <http://www.sciencedirect.com/science/article/pii/S1383762120301351>
16. Raravi, G., Andersson, B., Nélis, V., Bletsas, K.: Task assignment algorithms for two-type heterogeneous multiprocessors. *Real-Time Syst.* **50**, 87–141 (2014). <https://doi.org/10.1007/s11241-013-9191-3>
17. University, P.: Princeton application repository for shared-memory computers (PARSEC). <http://parsec.cs.princeton.edu>



Boosting Performance in Parallel Computing Models with a New Experimental Architecture

Alberto Arteta Albert¹(✉), Akshay Harshakumar¹,
Luis Fernando de Mingo López², and Nuria Gómez Blas²

¹ Department of Computer Science, Troy University, 112C McCall Hall (MSCX),
Troy, AL 36082, USA

{aarteta,aharshakumar}@troy.edu

² Departamento de Sistemas Informáticos, Escuela Técnica Superior de Ingeniería de
Sistemas Informáticos, Universidad Politécnica de Madrid, Calle Alan Turing,
28031 Madrid, Spain

{fernando.demingo,nuria.gomez.blas}@upm.es

Abstract. The von Neumann architecture, often called the Traditional architecture, describes the design for an electronic digital computer as having five main components: processing unit, control unit, memory, external storage, input/output mechanisms. The processing unit interacts with the rest of the components through the system bus. That helps in carrying the data as well as the addresses. Over the years, that architecture has evolved with the advancement in technology. Today we find that we face many constraints regarding the deployment of some hardware and software features developed in recent years due to the inherent nature of the traditional architecture. This paper shows how to use this architecture to boost performance by introducing a variation in signal processing by placing a different memory type that reduces stall operations when processing different parallel threads.

Keywords: Experimental architecture · Parallel computing · Dynamic hardware

1 Introduction

One such prominent issue is the constraint between address bus size and maximum available memory, and another one being the disparity in latencies of accessing main memory and storage media. A recent development in this field has led to the design of a Dual-Space Storage (DSS) technology that utilizes the non-volatility of NVRAMs [8, 10], combined with its random-access capability to design a memory architecture that bridges the gap between internal and external memory. Alas, this model tends to fail in practical applications and is not a universal solution to the issue at hand. This paper introduces a Hybrid Memory Architecture based on the memory shift technique of DSS technology that

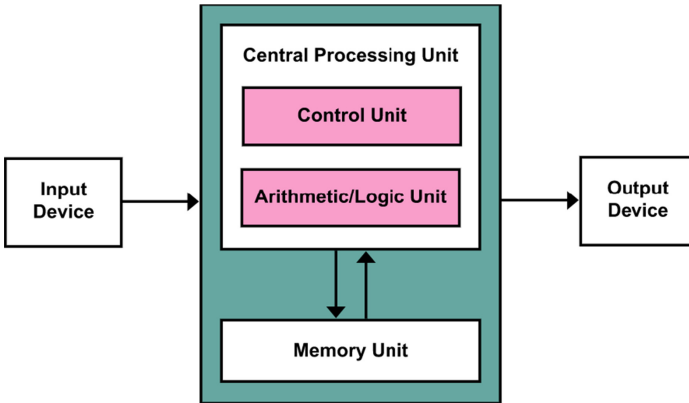


Fig. 1. The block diagram of von Neumann architecture [13, 14]

combines both NVRAM and traditional DRAM to provide optimal performance and persistence simultaneously. This bridge between the two types of memories is an additional layer in the topology; however, it barely affects its performance. The trade-off is the enhancement of the existing features of the computer system module, namely Virtual Memory.

The Hungarian-American mathematician and computer scientist John von Neumann first introduced the idea of a stored-program digital computer in the 101-page document First Draft of a Report on the EDVAC in 1945 [13, 14]. The document pictures an electronic digital computer as having these five major components:

- A processing unit.
- A control unit.
- A memory unit that keeps data and instructions
- An external mass storage medium
- An input and output unit
- A system bus unit is connecting the above.

The System Bus can be sub-divided into a data bus, address bus, and control bus [9, 13].

Over the years, computer manufacturers have used variations of this architecture to design systems that have served their purpose to this day [6, 7]. Many original architecture designs have been replaced or modified to form better or different system models to suit the increasing or varying needs and demands of modern computing [2, 3, 12]. Through-out all these technological advancements, some key concepts of the original architecture have endured and survived to this day [5]. Below there are the principal two:

- The existence of the main memory, henceforth called memory, temporarily stores and currently executing instructions and working as data and a storage medium, and virtual memory, henceforth called storage, separate from the memory that stores large-scale data in a permanent fashion.
- The use of an address bus, which is part of the system bus, relays the location of resources needed for processing data.

These two key features of traditional architecture have been facing many constraints in recent years, with it only going to worsen over the coming years. In particular, memory is used to store instructions and data that are being executed; many of these constraints have been related to power management, heat dissipation, and intractability [4].

2 Signalling

Memory in modern computer systems is random-accessible; that is, they can be accessed in a non-sequential manner by specifying the row and column where the data lies [11]. They are hence speedy and are called Random-Access Memory or RAM. The RAM in computers is volatile; they lose the data held once the power has been switched off. Hence, they cannot be used to store data of a permanent nature that needs to be accessed across an extended period, preferably through multiple on-off cycles. Another drawback of RAM is that this type of memory is costly. These factors mean that a storage medium for permanently storing large amounts of data is an essential part of a computer system. That is why we still have the concept of internal and external storage. An HDD, often used as external storage, can cost about a hundred times as cheap as a RAM. The existence of an external memory separate from the internal memory comes with its share of problems. The difference in speed between these two tends to affect the system's performance in question, with a significant deterioration in performance observed when applications requiring large amounts of memory forces the system to access secondary memory more often. Another example is the noticeable slowing down of applications when the virtual memory is accessed since virtual memories are traditionally allocated in external storage media [1].

The next key feature of traditional computers is the address bus's role in being the communication medium between the processor and the memory. The address bus in a system has n -lines used to carry n -bits across the processor and memory. These n -bits make up a cell's address containing a byte of data in memory, hence the name byte-addressing. Suppose the memory has L cells capable of storing a byte of data each, addressed from 0 to $L-1$. That means that the n -bits in the address bus must carry the numbers 0 to $L-1$, which roughly translates to the fact that the address bus's largest address is $2n$, which should equal $L-1$ in value. If the memory has a less than L capacity, say M , it still would work since the addresses M to $L-1$ would be left unused by the address bus. The problem arises when we decide to increase the available memory to a value greater than L . The existing address lines would not be able to address any cells over $L-1$. That is

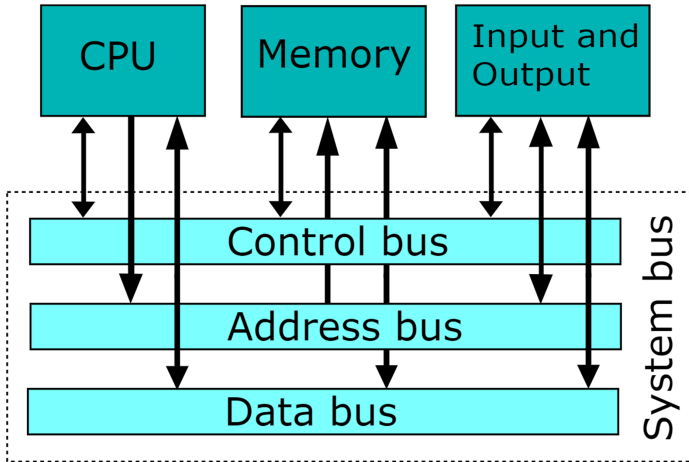


Fig. 2. The block diagram of a system bus [13,14] in a conventional architecture

the infamous bottleneck between address bus size and maximum memory size. A system that has n -lines can accommodate a memory module of at most $2n$ bytes. Older systems used to have address lines of 32-bits, which meant they could hold a memory of size 232 bytes, or 4 GB. The 4 GB threshold was hit a long time ago, and current systems have advanced to 64-bit addressing due to this very reason. A 64-bit address bus can handle 264 bytes, or 16 EB of memory, which would be more than sufficient for many years to come. Although the use of 64-bit addressing over 32-bit addressing may seem like a perfect solution, it needs to be pointed out that only medium and large-scale systems find it viable. Small systems that need more significant memory than 4 GB yet not willing to switch over to 64-bit addressing still find issues with this bottleneck. That is where the memory shift technique, detailed in the next section, used in DSS systems come into play [10].

To solve these issues, the CPU does extra work by generating logical addresses and mapping. So the proposed model includes an experimental design, a Hybrid Memory Architecture (HMA) with bus communicator between the organizational units, specifically between the CPU and the new topology of the memory.

This new architecture addresses the speed difference between internal and external memory and the address bus issue, limiting the maximum memory capacity. HMA uses a combination of traditional RAM and Non-Volatile RAM (NVRAM) to bridge the gap between internal and external memory. It also uses the memory shift technique to eliminate the cap on memory size due to the address bus. HMA draws mainly from the DSS technology proposed by Jin Yi et al. in 2013 [8,10].

Section 3 of this paper discusses the DSS technology and its advantages and disadvantages. Section 4 concentrates on the principles of traditional RAM types, NVRAM features, and the concept of virtual memory. Section 5 defines the fun-

damental concepts of the HMA model and draws out its advantages over the DSS model.

3 Fundamental Principles of a Dual-Space Storage System

The Dual-Space Storage principle was introduced in 2013 by Jin Yi et al. when his research team applied for the Chinese Invention Patent “A Computer System and Data Reading and Writing Method” [10], and then published two papers [8, 10].

The keystone of a DSS system is using an NVRAM to serve as the basis for its memory usage. The key concept is that both memory and storage are being merged into a single entity, in this case, an NVRAM, that serves the purposes of both traditional internal and external storage. The process of merging is twofold. On the one hand, the random-access capability of an NVRAM is used to substitute a traditional RAM. On the other hand, the persistent nature of the NVRAM is used to provide external storage capabilities. That creates a partition within the system, namely, word-space and block-space. The word-space is the space within the DSS system that provides internal storage functionality since it is accessed byte-wise or word-wise and emulates a RAM. The DSS system’s block-space portion is accessed by blocks, which is more like an external storage media. Together, the word-space and block-space make up the DSS system.

A DSS system primarily aims at abruptly increasing the total amount of memory random-accessed by the system to a size that rivals an external memory module. The system would then have an area of dual-space storage that can work as both internal and external storage at the same time. That does not mean that the CPU addressing system can suddenly address all this extra space since that value remains at $2n$, for n -lines of addressing bits. That is where the concept of memory shift comes into play.

Figure 3 shows the schematics of a DSS system that uses memory shifting to increase memory addressable by a system bus. Suppose the address bus has n -lines, and the DSS system is fixed to a size of $2m$, m being the number of cells in the memory, called Window walls. The address lines coming from the CPU are divided into two groups: the low-bits and the high-bits. If there are n -lines in the CPU bus, s lines are considered the low-bits, and the remaining $n-s$ lines are considered the high-bits. The s low-bits are directly mapped to the DSS module. The remaining $n-s$ high-bits are given to the latch group that contains $2n - s$ latches. To address the DSS module, a total of m bits are required. S bits are initially obtained; the remaining $m-s$ bits are obtained from the latches. Each latch is preset with $m-s$ bits by the system, although this is not permanent and keeps changing according to the physical memory that needs to be addressed. Once a latch is selected, the $m-s$ bits stored in the latch become the high-bits of the final physical address, while the originals low-bits are used as such. Hence, together they make up m bits for addressing. The mapping window, window frames, and the latch values are controlled through the system management module, which is not depicted in the schematics.

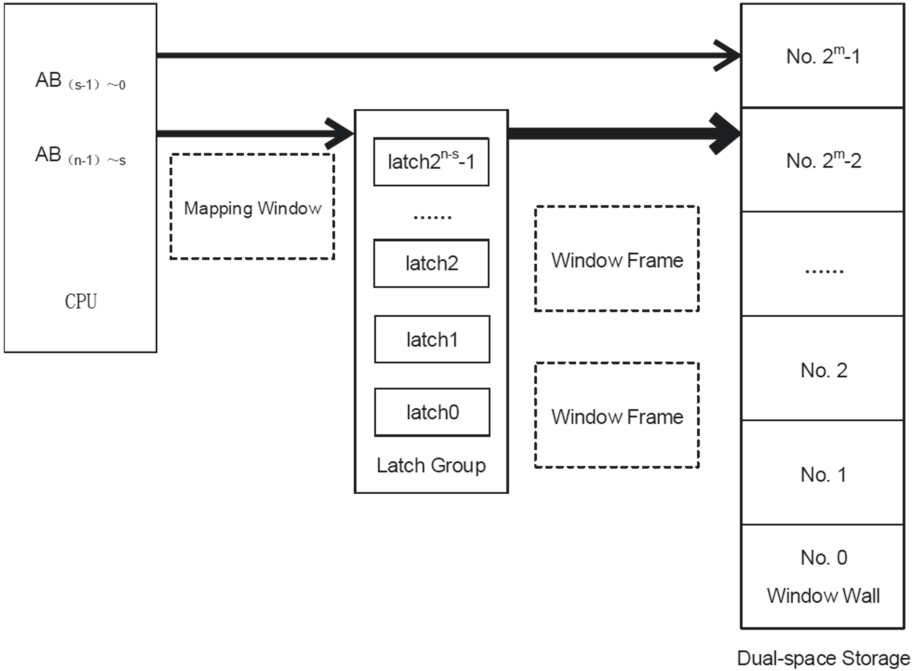


Fig. 3. The schematic diagram of a Dual-Space Storage system using the Memory Shift technique [8]

4 Working on a Dual-Space Storage Prototype

Jin Yi et al. have set up a DSS prototype to verify the proper working of a DSS based system. The general set up of the prototype is illustrated in Fig. 4.

The Experiment uses an embedded processor, S3C2440 of TianQian Company, that has a 21-line address bus. The DSS module comprises four NVRAMs, Norflash chips S29GL01G of Spansion Company, 256 MB each, giving a total of 1 GB dual-space storage. In a traditional system, a 21- line address bus would translate to a 2 MB memory access (2^{21} bytes = 2 MB). A 2 MB address is being mapped to a 1 GB dual-space storage in this prototype, requiring 30 address lines. Other components used in the setup are 74ALS138, a 3–8 decoder, and a shift latch 74AC16373DL.

The lower 18 address bits are taken as the s low- bits of the DSS system, which leaves $n-s$ as 3 of the higher address bits. These three higher bits are plugged into a 3–8 decoder 74ALS138. Once the 3–8 decoder activates one of its eight outputs, it is further connected to the shift latch group 74AC16373DL with eight latches. Each of these latches is preset, with 12 bits used to make up the address’s higher 12 bits. Together with the original 18 lower address bits, these make up the 30-bit address bits required for accessing the chips of S29GL01G. If the DSS system has multiple chips in the DSS module, as is the case in this

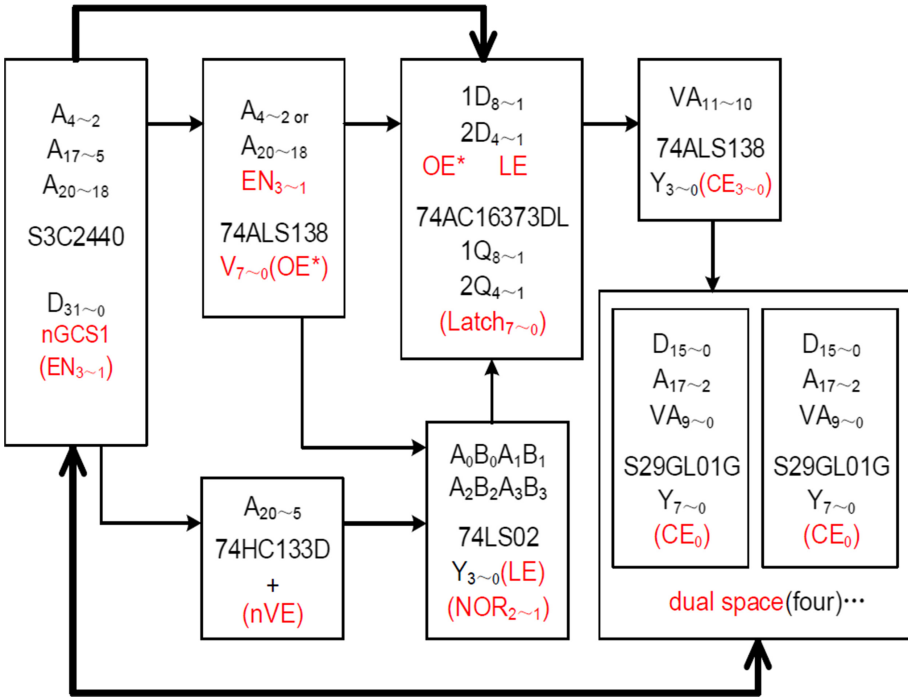


Fig. 4. The schematic diagram of a Dual-Space Storage system using the Memory Shift technique [8]

prototype, then the higher bits can be further passed through a suitable decoder to select the chip through indirect addressing. In this case, two of the higher bits are used for switching to the right chip within the DSS system. A 2–4 decoder can achieve this functionality, but another 3–8 decoder (upper-right decoder in Fig. 4) is used in this setup to maintain the uniformity of components used.

The most important part of the DSS system is the realization of mapping window shift and window frameshift. The system achieves this management module, shown as nVE and NOR blocks in Fig. 4. Mapping window shift is when latches go through different window frames and maps to a physical address in the window wall. The system sets these values in the system management module through the use of vectors. When the same latch maps to a different window frame by changing the latch’s value, it is called “window frameshift.” Together, both of these shifts help in realizing the memory shift technique in DSS systems.

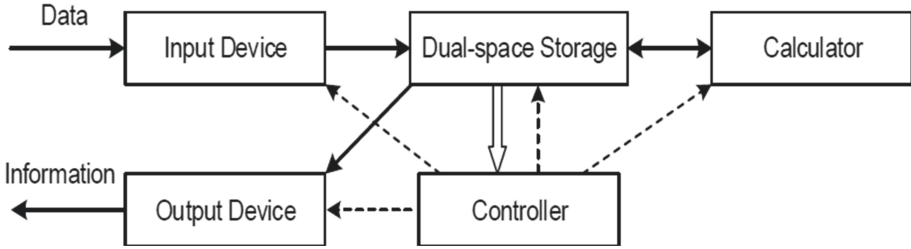


Fig. 5. A Dual-Space Storage system based computer architecture [8]

5 Drawbacks of Dual-Space Storage Model vs. the Proposed HMA

The DSS model helps eliminate two critical problems faced by the von Neumann architecture described in earlier sections. Despite these triumphs, the DSS model fails to fulfill specific requirements in different classes of computer models.

1. **Reduced speed of NVRAMs:** The DSS model has successfully increased the total amount of memory that can be random-accessed by a computer, but to do so, it has chosen to use NVRAMs, which are slower than DRAMs used in traditional systems. That means that a specific reduction in applications' performance can be observed when used in such scenarios. Applications or systems that expect a fast response from the processor may not necessarily seek memory-intensive operations would feel a drop in certain operations' response times. To circumvent this issue, an improved model needs to make use of DRAMs' speed and the persistence of NVRAMs. The HMA can setup a priority scheme for applications and redirect to DRAM the addresses data and Instructions of those application process that require faster management. DSS is unable to provide that service.
2. **Increased cost and decreased durability of NVRAMs:** NVRAMs are a general class of non- volatile persistent memory devices that mostly use flash memory to achieve non-volatility. Flash memory, and by extension, NVRAMs, are much costlier than the already expensive DRAMs. NVRAM technology is still at an infant stage, and hence the number of manufacturers and R&D regarding NVRAMs are few and far between. Since present NVRAMs are furnished out of flash drives, they are prey to the already pressing disadvantages of flash drives, namely its durability in terms of the number of read-write cycles. Currently marketed NVRAMs cost about 30USD per GB, roughly three times that of regular DRAMs. A model that uses a hybrid of NVRAMs and DRAMs are thus preferable to a DSS system functioning solely on NVRAMs.
3. **Overall reduction in storage capacity:** The DSS system that uses dual-space is significantly limited in its overall storage capacity. While the memory has certainly been increased and completely random-accessed for an increase in speed, systems, or applications that are not performance-oriented, do not benefit from it. Systems that require storage in bulk are now left with what

Table 1. Comparative of three architectures in a sequential and a parallel system of 2, 3, and 4 cores

	TRA	DSS (sim1)	HMA (sim2)
1	0.157/0.376	0.154/0.378	0.187/0.412
2	0.191/0.351	0.251/0.284	0.263/0.312
3	0.208/0.302	0.266/0.275	0.278/0.292
4	0.225/0.291	0.271/0.267	0.287/0.278

little of the dual-space is left for external storage. That may have an adverse effect if used in specific systems like database servers where storage and performance must go hand-in-hand and where one must not eclipse the other by a considerable margin. That would indicate that the complete removal of external storage media may not be the best possible solution for a system that can pan out multiple usage areas. The Hybrid approach does not technically address this issue. However, it provides an smarter management.

4. Latency increase due to memory shift operations in NVRAM: As per the current designs, all the metadata and operations required for the memory shifting operations in DSS systems are done using memory from the NVRAM. That means that system management takes place from within the NVRAM itself. The memory shift operations, manipulations of DSS meta-data, word-space, and block-space operations are all based on the NVRAM. The NVRAM, being slower than traditional DRAMs, will be slower in resolving the addressing operations it needs to perform, owing to its slower storage. That would lead to a further reduction in performance when memory shifting operations are invoked. That is resolved by using a hybrid solution of a distributor module that balance the use DRAMs and NVRAMs. This balancer proposed follows the resource based adaptative approach for the DRAM and NVRAM.

Based on these points, it is evident that a better model needs to be developed to address these issues. Some of the critical features of this new model are:

- A hybrid memory module that hosts both DRAM and NVRAM chips to draw from both technologies’ advantages, hence eliminating the drawbacks of using any one of these.
- Retention of an external storage media to ensure the availability of substantial storage spaces while moving performance-conscious operations, like Virtual Memory allocation, to the hybrid memory module.

The Table 1 shows a comparison of the impact in stalls and latency of the three different architectures: Traditional (TRA), Dual space storage (DSS), and the proposed Hybrid model (HMA).

The metrics in every cell are normalized stalls/normalized latencies. The values have been collected with a trace of the performance monitor in the traditional architecture, and generated automatically in a simulation of DSS and HMA. The

simulation has been taking into account the penalty of the communication setup between the memory units and CPU.

6 Conclusion

In this paper, a new model based on the DSS model has been proposed. The newly proposed HMA model eliminates the performance issues of using NVRAM alone by multiplexing it with volatile RAM modules for better speed, preserving the conductivity, and allow the CPU to boost the Cycles per second without damaging the architecture and reducing the latency. This particular proposal could work well in new models where the communication modules could add the proposed layer for ensuring decent performance in parallel systems. On the other hand, the idea is experimental and could also significantly affect the pricing and not guarantee success. More studies need to be done to integrate crystal materials in the new computing era and experimental topologies. Nevertheless, the retention of external storage has eliminated bottlenecks in the storage capacity of systems. Moreover, the traditional system of allocating Virtual Memory has been altered to offer a better solution with a bridge layer in the DSS model to increase the communication in parallel processing architectures. Therefore, increasing the cycles per second and therefore impacting the latency.

References

1. Amdahl, G.M.: Validity of the single processor approach to achieving large scale computing capabilities. In: Proceedings of the April 18–20, 1967, Spring Joint Computer Conference. AFIPS 1967 (Spring), pp. 483–485. Association for Computing Machinery, New York (1967)
2. Asanovic, K., et al.: The landscape of parallel computing research: a view from Berkeley. Technical report UCB/EECS-2006-183, University of California, Berkeley, USA (2006)
3. Barney, B.: Introduction to parallel computing. Technical report UCRL-MI-133316 (2014)
4. Brooks, F., Brooks, F., Education, P.: The Mythical Man-Month: Essays on Software Engineering. Addison-Wesley, Boston (1995)
5. Godfrey, M.D., Hendry, D.F.: The computer as von Neumann planned it. IEEE Ann. Hist. Comput. **15**(1), 11–21 (1993)
6. Hennessy, J., Patterson, D., Larus, J.: Computer Organization and Design: The Hardware/Software Interface. Morgan Kaufmann Publishers, Burlington (1998)
7. Hennessy, J.L., Patterson, D., Goldberg, D., Asanovic, K.: Computer architecture: a quantitative approach. In: The Morgan Kaufmann Series in Computer Architecture and Design Series. Morgan Kaufmann Publishers (2003)
8. Macdonald, J.R.: Interpretation of AC impedance measurements in solids. In: Mahan, G.D., Roth, W.L. (eds.) Superionic Conductors. Physics of Solids and Liquids, pp. 81–97. Springer, Boston. https://doi.org/10.1007/978-1-4615-8789-7_6
9. Null, L., Lobur, J.: The Essentials of Computer Organization and Architecture. Jones and Bartlett Publishers Inc., Burlington (2006)

10. Ouyang, S., Peng, J., Jin, Y., Shen, Y., Liu, X., Li, W.: Structure and theory of dual-space storage for ternary optical computer. *SCIENTIA SINICA Informationis* **46**(6), 743–762 (2016)
11. Rabaey, J.: Digital integrated circuits: a design perspective. In: Prentice Hall Electronics and VLSI Series. Prentice Hall (1996)
12. Rauber, T., Runger, G.: Parallel Programming: for Multicore and Cluster Systems. Springer, Heidelberg (2013). <https://doi.org/10.1007/978-3-642-37801-0>
13. von Neumann, J.: First draft of a report on the EDVAC. Technical report, Digital Press (1981)
14. von Neumann, J.: First draft of a report on the EDVAC. *IEEE Ann. Hist. Comput.* **15**(4), 27–75 (1993)



RRW: A Reliable Ring Waveguide-Based Optical Router for Photonic Network-on-Chips

Meaad Fadhel , Lei Huang, and Huaxi Gu  

State Key Laboratory of Integrated Service Network,
Xidian University, Xi'an, China
meaad_fadhel@yahoo.com, leihuang0694@163.com,
hxgu@xidian.edu.cn

Abstract. Previously, Photonic Network-on-Chip (PNoC) was proposed to solve the serious bottlenecks experienced by the traditional Electrical Network-on-Chips (ENoCs) such as End-to-End (ETE) delay, bandwidth and energy consumption. Therefore, it has attracted significant interest in the past few years. Optical Routers (ORs) are the essential components of a PNoC. One of ORs' major challenges is reliability. Optical routers are constructed by some optical devices, such as waveguides, and Microring Resonators (MRs). If one of these components; especially MRs, suffer from a breakdown due to thermal or tuning issues, the entire network will breakdown as well. Some optical routers and routing algorithms have been proposed to overcome such dilemma. However, in this paper we propose a universal method that can be implemented in any optical router in order to increase its reliability, without exposing it to additional contention or blocking issues. We implement a ring waveguide with a limited number of MRs to any router to provide a backup path for any faulty port-to-port communication. This method does not affect the normal flow of signals within the original router. To evaluate the efficiency of this method we implemented it into two known non-reliable optical routers, and compared them with two existing reliable optical routers. The results show that optical routers based on our method enjoy the least worst-case insertion loss, and crosstalk compared to the other two reliable routers.

Keywords: Optical Network-on-Chips · Reliable optical routers · Non-blocking routers · Interconnections · Photonic routers

1 Introduction

Network-on-Chip (NoC) has always best presented the features of Integrated Circuits (ICs) technologies. However, the number of cores and processors in a single chip, had rapidly increased in accordance with Moore's Law [1]. This impaired the efficiency of the traditional NoC and emphasized the need for a new technology that can meet such high communication requirements [2]. Thus, Photonic Network-on-Chip (also known as Optical Network-on-Chips, ONoCs) was proposed later on, to accommodate communication demands within the power consumption limits [3]. PNoC enjoys less delay, higher bandwidth, less power dissipation, less crosstalk and electromagnetic interface compared to the traditional ENoC [4, 5].

On-chip interconnections consist of essential building blocks, such as topologies and routers. Many topologies were reported in the literature, such as mesh, torus [6], and many others [7, 8]. A major component of PNoCs is the optical router. Routers' job is to send and route packets to the direction of their destinations. Thus, it determines the accuracy and efficiency of the communication. Since the first appearance of PNoC, several optical routers have been proposed, such as the routers in [9, 10, 11].

With the smaller chip sizes, ICs have become more variation susceptible. Thus, more likely to endure faults encountered by some manufacturing defects or operating frequency. Moreover, due to component failures, smaller ICs are more prone to transient faults. Unlike ENoCs, PNoCs are susceptible to aging and production variations more than transient faults [12–14]. Aging is more likely to happen to active components or elements with high thermal variation [15]. Thus, active components, e.g. MRs, suffer from a higher failure rate compared to passive components, e.g. waveguides [15]. This endangers the reliability of communications in on-chip interconnections. Critical mission applications requires high reliability, because the system in such applications must function correctly under any circumstances. Thus, the reliability of an optical router is a hot topic nowadays [16–19], since ORs are one of the main active components of PNoCs.

An optical router loses its efficiency when a break down occurs to any of its components. To solve this problem some authors have investigated rerouting the optical signal [16, 17]. One downside of this solution is that it often reroutes traffic in a similar fashion that causes traffic in one area, and ignores the faulty routers all together. Other authors focused on the thermal variation issues. Authors in [18] proposed a low-power thermal-resilient ONoC (RONoC) to alleviate faults, as well as investigated the thermal variation of on-chip power distribution. Another way to solve this problem is by using redundant MRs. Authors in [13] proposed a new fault-tolerant low-power 3D optical router, called FTTDOR, for PNoCs. They used some redundant MRs in critical locations to overcome reliability issues. Moreover, the authors in [19] designed a highly reliable OR structure for PNoCs. They increased the alternative restore paths by adding small hardware redundancy to their previous OR, to guarantee that the OR can still maintain normal communications. Moreover, they claimed that their node reuse fault-tolerant algorithm (known as FTRA-NR) can find the best restore path within each faulty OR. Although, both of the designs in [13] and [19] increase the number of MRs for reliability purposes, they cause extra contention and insertion loss, since they use more resources (eg. MRs) to reroute faulty signals in many cases.

This paper proposes a universal method which can be implemented to any optical router for reliability purposes, without exposing it to some contention or blocking problems. This method implements a Reliable Ring Waveguide (RRW) with a limited number of MRs to any optical router to provide a backup path for any faulty port-to-port communication. Moreover, signals transmitted using the backup path do not affect the transmitted signals within the default path.

This paper illustrates the proposed RRW architecture and implementation in Sect. 2. Section 3 evaluates the proposed method compared to two other 7×7 reliable optical routers for 3D mesh; moreover, analyzes the insertion loss and crosstalk noise of all routers. In Sect. 4, we conclude this paper and discuss our future work.

2 Architecture Overview

2.1 RRW Architecture and Implementation

Optical routers are constructed by waveguides, which act like wires in traditional NoCs, and Basic Switching Elements (BSEs), which are basically one or two MRs connecting two parallel or crossed waveguides. Signals are switched from one waveguide to the other depending on the state of the MR. If the MR is on an ON-state, the MR will switch the signal passing by, from the current waveguide to the other. In contrast, if the MR is on an OFF-state, the signal will pass through the MR along the same waveguide without being switched, as shown in Fig. 1(a). Thus, faults occurring at one of these devices result in major misrouting issues and data loss.

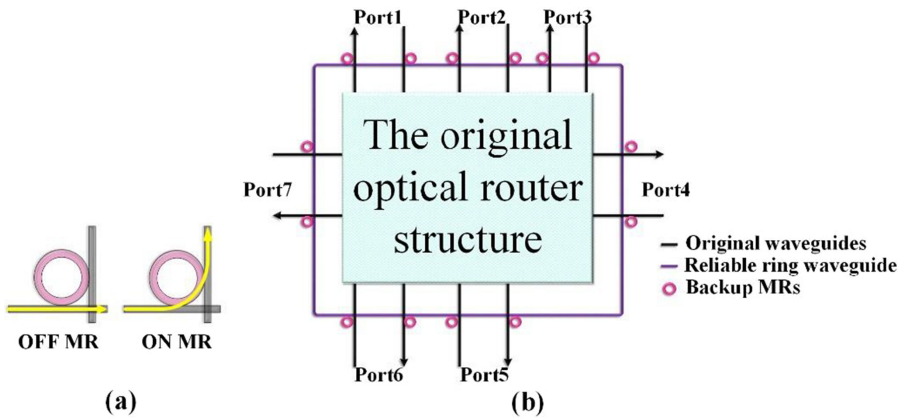


Fig. 1. (a) OFF and ON states BSE, (b) the reliable ring waveguide architecture.

The reliable ring waveguide architecture proposed in this paper, guarantees that such faults will not require changing the route, but simply uses a backup path within the same OR. The method proposed can be realized by implementing a single ring waveguide and some MRs to any OR. The ring waveguide works as a backup path in case any of the original router components breaks down for any reason. As shown in Fig. 1(b) the ring waveguide must be located at the beginning and the end of every input and output port, respectively. This helps to avoid any conflict with the original design and simplify the method. Moreover, the MRs are located at every waveguide crossing that is formed by the intersections of the ring waveguide and the input or the output waveguides as depicts in the figure. At input ports, the MRs are located on the left side of the input waveguide to form an add point to the ring waveguide [20]. On the contrary, at output ports, the MRs are located on the right side of the output waveguide to form a drop point to the ring waveguide [20]. In addition, the number of MRs added by the RRW is $2N$ for every $N \times N$ optical router.

Fig. 2(a) presents an example of a non-reliable 7×7 optical router, proposed in [20], whereas Fig. 2(b) shows the same OR after implementing our method. As shown,

the ring waveguide does not affect the functionality or connections of the original optical router, yet increases the path diversity. However, it adds some insertion loss and crosstalk to the original router, which is a slight increment compared to other reliable optical routers, as will be illustrated in the following section.

In case of component failure, the router will use the ring waveguide as a backup path to transmit data, instead of the default path. Moreover, one signal is allowed to use the ring waveguide at a time. In other words, the reliable ring waveguide can only realize one faulty signal at a time. Finally, other communications on the default path will never be affected by the communications on the backup path.

Given that the original optical router is strictly non-blocking, the only blocking that could occur, would be either within the ring waveguide or at its intersections with the original router. However, our architecture makes sure that it is implemented only at the beginning and end of the port, with an add point at the beginning of the input port and a drop point at the end of the output port. Thus, it guarantees that the add points will always be in front of the drop points to prevent overlapping [20].

For most faults, this design provides an alternative path. Each communication pair in the router has two alternative paths to exchange data. Thus, provides higher path diversity for reliability purposes.

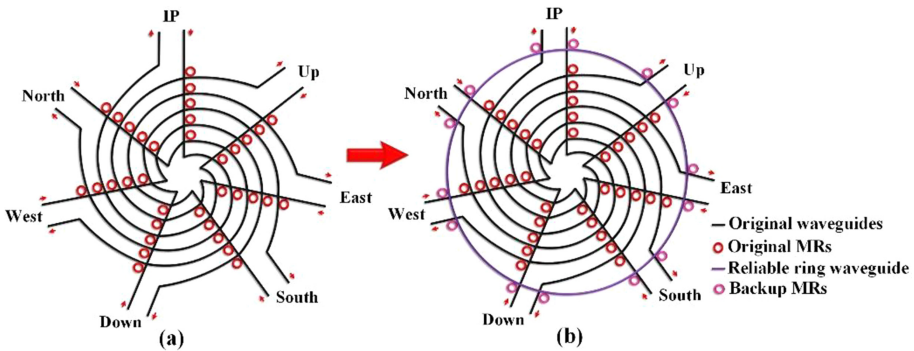


Fig. 2. An example of implementing the reliable ring waveguide to the 7×7 optical router in [20].

2.2 Communication Mechanism in RRW

In this subsection we demonstrate the communication mechanism of our architecture. As previously mentioned, our architecture allows each router to function normally. Thus, we, here, illustrate the signals flow within the RRW. Signals can be either propagated clockwise or counter clockwise, depending on the location of the MRs with regard to the input or output port. Since we previously mentioned that MRs are located at the left side of the input port and the right side of the input port, the flow of signals should be clockwise as shown in Fig. 3.

The figure shows an example of an optical signal transmitted from East to Down using the default path, whereas another signal is transmitted from Up to West using the

backup path. Both communications are independent and the backup path does not affect the normal flow of signals within the original router. The figure shows that in order to transmit an optical signal in the RRW, two MRs at most are required to be turned on to realize the transmission.

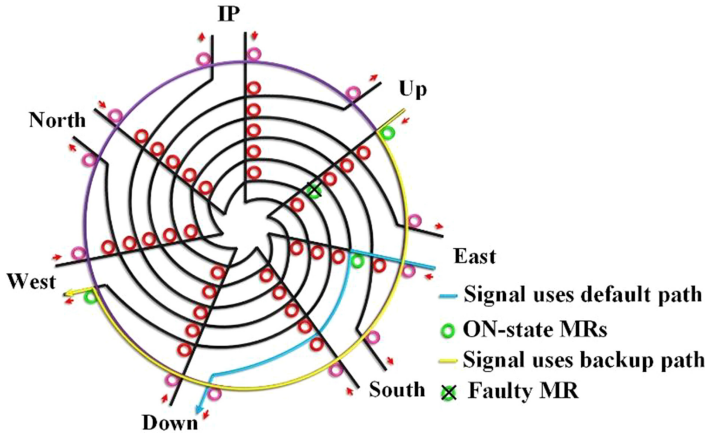


Fig. 3. An example of communications in both the default and the backup paths in an RRW-based OR.

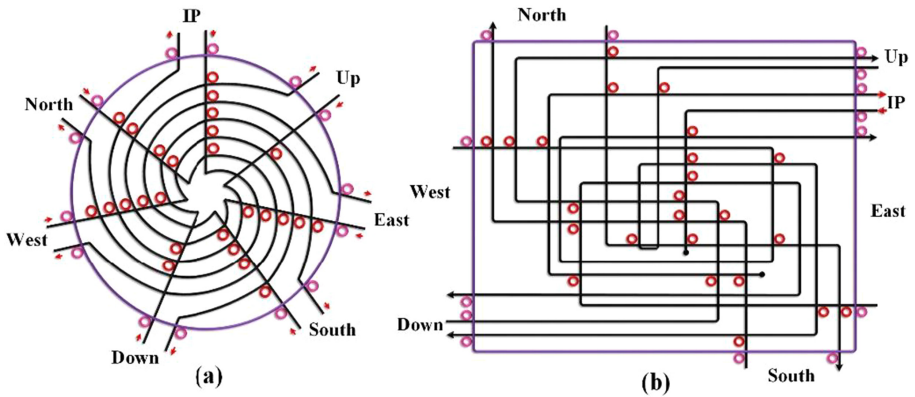


Fig. 4. RRW-based optical routers (a) RO-Uni. (b) RO-Votex.

3 Evaluation and Analysis

To evaluate the performance of our architecture we implement the reliable ring waveguide into two 7×7 different optical routers proposed in [20] and [21], and compare them with 7×7 FTTDOR proposed in [13], and the 7×7 NRFT optical router in [19]. Since the latter two reliable routers are based on XYZ routing algorithm, whereas the universal

router in [20] and Votex in [21] are fully connected routers. In XYZ routing algorithm, signals are transmitted along the X dimension (i.e. West or East ports) until no further move is needed in this dimension, then along Y dimension (i.e. North or South ports), and finally along Z dimension (i.e. Up or Down ports). Thus, no communications are established from Z dimension to X or Y dimensions, nor from Y dimension to X dimension in XYZ-based ORs. On the other hand, fully connected routers connect all ports together without prioritizing some dimensions over the others; i.e. all ports can communicate with all ports. Therefore, we optimized the optical routers in [20] and [21] by reducing the unused MRs for the unused communications, for fair comparisons, before implementing our method into both of them. We reduced the MRs responsible for communications between the following pairs: Up/Down \rightarrow North/South/East/West, and North/South \rightarrow East/West. The architecture of the optimized optical routers with the implemented RRW are shown in Fig. 4(a) and (b).

3.1 Insertion Loss

When a signal is transmitted in a waveguide, it faces different problems (e.g. light scattering and leakage). Another essential loss is caused by passing through optical devices, such as MRs, waveguide crossings and bends, which affects the scalability of the whole network. Thus, using the parameters in Table 1, we evaluate the Insertion Loss (IL) of each optical router and show the worst-case IL. To evaluate the backup path of each router, we assume that only one MR fails at a time.

Table 1. The parameters of different optical devices insertion loss and crosstalk coefficient.

Parameters	Value	Unit
MR drop loss	-0.5	dB [22]
MR through loss	-0.005	dB [22]
Waveguide bend loss	-0.005	dB/90° [23]
Waveguide crossing loss	-0.04	dB [24]
Propagation loss	-0.274	dB/cm [24]
On-state MR crosstalk coefficient	-25	dB [22]
Off-state MR crosstalk coefficient	-20	dB [22]
Waveguide crossing crosstalk coefficient	-40	dB [24]

Table 2 presents the maximum, minimum, and average port-to-port IL of the four optical routers. The total insertion loss in each column is given by:

$$IL_{total} = \sum (IL_{bend} + IL_{cross} + IL_{MR-off} + IL_{MR-on} + IL_{prop}) \quad (1)$$

Where IL_{bend} is the insertion loss presented by waveguide bends, IL_{cross} is the insertion loss presented by waveguide crossings, IL_{MR-off} is the insertion loss of

passing through an OFF MR, IL_{MR-on} is the insertion loss of tuning with an ON MR, whereas IL_{prop} is the insertion loss introduced while propagating along the waveguide.

NRFT architecture uses two separated optical routers, which are a 6×6 OR and a 3×3 OR for intra and inter-layer communications, respectively. Thus, the minimum port-to-port IL (-0.12 dB) would be in the smaller router from Up port to Down port and vice versa. Similarly, the minimum insertion loss in FTTDOR is (-0.115 dB), which is the one from Up port to Down port as well, since the UP and Down waveguides are only connected to themselves and the IP core and not to others. Nevertheless, the minimum insertion loss of the RRW-based optimized universal router (RO-Uni) is -0.53 dB for both the default and backup paths, and the minimum IL of the RRW-based optimized Votex (RO-Votex) is -0.59 dB and -0.73 dB for the default path and the backup path, respectively. Although the first two routers may have the minimum insertion loss, routers using our method enjoy the least average and worst-case insertion losses. RO-Uni has the least average IL (-0.75 dB) in the default path and a -1.101 dB IL for the backup path, whereas the average ILs of RO-Votex are -0.87 dB and -1.127 dB for the default and backup paths, respectively. The average IL of FTTDOR is low as well, with a -0.76 dB and -0.98 dB for the default and backup paths, respectively. However, NRFT has the highest average IL for both the default and backup paths, with a -0.93 dB and -1.48 dB, respectively. In terms of the worst-case IL, RRW-based routers enjoys the least IL, with a -0.955 dB and -1.36 dB in RO-Uni and a -1.28 dB and -1.45 dB in RO-Votex for the default and backup paths, respectively. On the other hand, the worst-case ILs in FTTDOR are -1.365 dB and -2.245 dB and in NRFT are -2.185 dB and -2.705 dB for the default and backup paths, respectively.

Table 2. The insertion loss Comparison of optical routers based on our method and FTTDOR and NRFT for port-to-port communications.

Routers	Maximum IL(dB)		Minimum IL(dB)		Average IL(dB)	
	Default	Backup	Default	Backup	Default	Backup
RO-Uni [20]	-0.955	-1.36	-0.53	-0.53	-0.75	-1.101
RO-Votex [21]	-1.28	-1.45	-0.59	-0.73	-0.87	-1.127
FTTDOR [13]	-1.365	-2.245	-0.115	-0.115	-0.76	-0.98
NRFT [19]	-2.185	-2.705	-0.12	-0.12	-0.93	-1.48

3.2 Crosstalk

While passing through waveguide crossings and MRs, a small part of the optical signal will be directed to an unwanted channel. The later will affect other signals passing through these channels and cause a crosstalk noise.

Using the parameters in Table 1, we analyze the crosstalk noise experienced by the routers. The crosstalk noise added to an optical signal when transmitted from the i^{th} port to the j^{th} port in the OR can be calculated as follows,

$$N_{i,j} = \sum_{m=0}^n P_m^0 K_{i,j,m} \quad i, j \in \{0, 1, 2, \dots, n\} \quad (2)$$

Where P_m^0 is the optical power injected into the input of the i^{th} port, and $K_{i,j,m}$ is the crosstalk noise coefficient presented by P_m^0 onto the optical signal traveling from the i^{th} port to the j^{th} port in the OR.

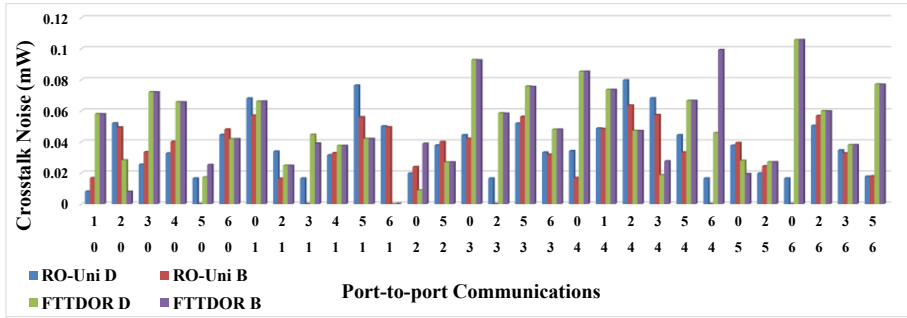


Fig. 5. Port-to-port crosstalk noise.

Fig. 5 shows the crosstalk noise of RO-Uni default and backup paths, and FTTDOR default and backup paths. The results show that RRW-based router reduces the worst-case crosstalk noise by 24.55% for the default path and 39.9% for the backup path. Moreover, the figure shows that RO-Uni experiences the least crosstalk noise in many communication pairs.

4 Conclusion and Future Work

Reliability of an optical router is a hot topic for researchers. It determines the efficiency and performance of the network. We proposed a universal method that is easily implemented to an optical router for reliability purposes. The method doesn't expose the optical router to further contention or blocking problems. In this method, we implement a Reliable Ring Waveguide (RRW) with a specific number of MRs, which is $2N$, to any $N \times N$ optical router to provide an alternative path for any faulty communication. Another important feature of the proposed method, is that signals transmitted using the alternative path do not affect the transmitted signals using the original path. Furthermore, the results of the numerical simulation show that RRW-based optical routers enjoy the least worst-case IL compared to FTTDOR and NRFT. Moreover, RO-Uni reduces the worst-case crosstalk noise by 24.55% and 39.9% for the default and backup paths, respectively, compared to FTTDOR.

In our future work, we will implement our method to various types of optical routers and investigate the reliability that RRW brings to them. We will further analyze insertion loss and crosstalk in network level as well as the power consumption and signal-to-noise ratio.

Acknowledgment. This work was supported in part by the National Natural Science Foundation of China under Grant 61634004 and 61934002, the Natural Science Foundation of Shaanxi Province for Distinguished Young Scholars under Grant No. 2020JC-26.

References

1. Wu, C., et al.: A multi-objective model oriented mapping approach for NoC-based computing systems. *IEEE Trans. Parallel Distrib. Syst.* **28**(3), 662–676 (2017)
2. Baharloo, M., Aligholipour, R., Abdollahi, M., Khonsari, A.: ChangeSUB: a power efficient multiple network-on-chip architecture. *Comput. Electr. Eng.* **83**, 106578 (2020)
3. Meindl, J.: Interconnect opportunities for gigascale integration. *Micro, IEEE* **23**(3), 28–35 (2003)
4. Barwicz, T., Byun, H., Gan, F., et al.: Silicon photonics for compact, energy-efficient interconnects [Invited]. *J. Opt. Networking*. **6**(1), 63–73 (2007)
5. Abdollahi, M., Mohammadi, S.: Insertion loss-aware application mapping onto the optical Cube-Connected Cycles architecture. *Comput. Electr. Eng.* **82**, 106559 (2020)
6. Shacham, A., Bergman, K., Carloni, L.P.: Photonic networks-on-chip for future generations of chip multiprocessors. *IEEE Trans. Comput.* **57**(9), 1246–1260 (2008)
7. Yang, W., Chen, Y., Huang, Z., Zhang, H., Gu, H., Yu, C.: A survey of multicast communication in Optical Network-on-Chip (ONoC). In: Shen, H., Sang, Y. (eds.) PAAP 2019. CCIS, vol. 1163, pp. 58–70. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2767-8_6
8. Guo, L., Hou, W., Guo, P.: Designs of 3D mesh and torus optical Network-on-Chips: topology, optical router and routing module. *China Commun.* **14**(5), 17–29 (2017)
9. Sun, S., et al.: Hybrid photonic-plasmonic nonblocking broadband 5×5 router for optical networks. *IEEE Photonics J.* **10**, 1–2 (2018)
10. Shi, X., Wu, N., Ge, F., Yan, G., Xing, Y., Ma, X.: Srax: a low crosstalk and insertion loss 5×5 optical router for optical network-on-chip. In: IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society, pp. 3102–3105 (2019)
11. Yahya, M.R., Wu, N., Fang, Z., Ge, F., Shah, M.H.: A low insertion loss 5×5 optical router for mesh photonic network-on-chip topology. In: IEEE Conference on Sustainable Utilization and Development in Engineering and Technologies (CSUDET), pp. 164–169 (2019)
12. Ye, Y., et al.: System-level modeling and analysis of thermal effects in WDM-based optical networks-on-chip. *IEEE Trans. Comput. Aided Des. Integr. Circ. Syst.* **33**(11), 1718–1731 (2014)
13. Meyer, M.C., Ahmed, A.B., Okuyama, Y., Abdallah, A.B.: FTTDOR: microring fault-resilient optical router for reliable optical network-on-chip systems. In: Proceedings of the IEEE 9th International Symposium on Embedded Multicore/Many-core Syst.-on-Chip, pp. 227–234 (2015)
14. Datta, I., Datta, D., Pande, P.P.: Design methodology for optical interconnect topologies in NoCs with BER and transmit power constraints. *IEEE J. Lightwave Technol.* **32**(1), 163–175 (2014)
15. Hu, Z.S., Hung, F.Y., Chen, K.J., Chang, S.J., Hsieh, W.K., Liao, T.Y.: Improvement in thermal degradation of zno photodetector by embedding silver oxide nanoparticles. *Funct. Mater. Lett.* **6**(01), 1350001 (2013)

16. Loh, P.K.K., Hsu, W.J.: Design of a viable fault-tolerant routing strategy for optical-based grids. In: Guo, M., Yang, L.T. (eds.) *Parallel and Distributed Processing and Applications. ISPA 2003. Lecture Notes in Computer Science*, vol. 2745, pp. 112–126. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-37619-4_13
17. Xingyun, Q., Quanyou, F., Yongran, C., Qiang, D., Wenhua, D.: A fault tolerant bufferless optical interconnection network. In: *Eighth IEEE/ACIS International Conference on Computer and Information Science, 2009. ICIS 2009*, pp. 249–254. IEEE (2009)
18. Tinati, M., Koohi, S., Hessabi, S.: Low-overhead thermally resilient optical network-on-chip architecture. *Nano Commun. Netw.* **20**, 31–47 (2019)
19. Guo, P.: Fault-tolerant routing mechanism in 3D optical network-on-chip based on node reuse. *IEEE Trans. Parallel Distrib. Syst.* **31**(3), 547–564 (2020)
20. Min, R., Ji, R.Q., Chen, Q.S., Zhang, L., Yang, L.: A universal method for constructing N-port nonblocking optical router for photonic networks-on-chip. *J. Lightwave Technol.* **30**(23), 3736–3741 (2012)
21. Zhu, K., Gu, H., Yang, Y., Tan, W., Zhang, B.: A 3D multilayer optical network on chip based on mesh topology. *Photon Netw. Commun.* **32**(2), 293–299 (2016). <https://doi.org/10.1007/s11107-016-0627-2>
22. Chan, J., Hendry, G., Bergman, K., Carloni, L.: Physical-layer modeling and system-level design of chip-scale photonic interconnection networks. *IEEE Trans. Comput. Aided Des. Integr. Circ. Syst.* **30**(10), 1507–1520 (2011)
23. Xia, F., Sekaric, L., Vlasov, Y.: Ultracompact optical buffers on a silicon chip. *Nat. Photonics* **1**, 65–71 (2007)
24. Fusella, E., Cilaro, A.: PhoNoCMap: an application mapping tool for photonic networks-on-chip. In: *Design Automation and Test in Europe Conference and Exhibition (DATE)*, pp. 289–292 (2016)

Author Index

- Albert, Alberto Arteta 418
- Barbosa, Aaron 162
- Blas, Nuria Gómez 418
- Bossard, Antoine 12
- Cai, Hengyu 22
- Chen, Liyou 318
- Chen, Peixiang 247, 259
- Chen, Qiurui 68
- Chen, Xin 295
- Chen, Yanping 187
- Chen, Ying 295
- Chen, Yuzhong 259
- Cheng, Qian 200
- Cheng, Yiran 378
- Chui, Wenhao 104
- Cunningham, Rhys 236
- Das, Zinea 405
- de Mingo López, Luis Fernando 418
- Djidjev, Hristo N. 162
- Dong, Yichuan 355
- Fadhel, Meaad 429
- Fang, Gengfa 34, 115
- Fang, Zisen 355
- Fatema, Israt 115
- Feng, Shengzhong 355
- Feng, Xiangyang 282
- Fu, Pengming 104
- Fu, Xianya 68
- Gao, Chuanji 331
- Gao, Yongqiang 365
- Gu, Huaxi 429
- Hahn, Georg 162
- Hamad, Salma Abdalla 174
- Han, Lin 390
- Han, Pu 390
- Han, Yunlong 34
- Harshakumar, Akshay 418
- Hendry, Ben 236
- Hou, Zhengxiong 343
- Hu, Miao 224, 270
- Hu, Weifang 390
- Hu, Zhangfeng 331
- Hua, Qiang 318
- Huang, Jiale 1
- Huang, Jiangyi 247, 259
- Huang, Jie 140
- Huang, Lei 429
- Huang, Qianjing 187
- Jiang, Murong 104
- Ju, Jiachen 81
- Kobayashi, Hiroaki 378
- Komatsu, Kazuhiko 378
- Kong, Xiaoying 34, 115
- Li, Huan-Yi 200
- Li, Min 81
- Li, Pan 318
- Li, Wenrui 68
- Li, Xingli 343
- Li, Xiong 331
- Li, Yan 318
- Li, Yanjun 331
- Li, Yongnan 150
- Lin, JianZhuang 93
- Lin, Shiwei 34
- Liu, Ang 34
- Liu, Jianxin 81
- Liu, Peng 1
- Liu, Qian 81
- Liu, Xiao 212
- Liu, Zhanghui 247
- Liu, Zhentao 140
- Luo, Chenyu 270
- Luo, Shunchao 307
- Ma, Teng 295
- Ma, Zhuo 295
- Mahmood, Adnan 47
- Moulik, Sanjay 405

- Nepal, Surya 174
 Ning, Chengming 22
 Pelofske, Elijah 162
 Qi, Fumin 355
 Qing, Duzheng 68
 Sagar, Subhash 47
 Sang, Yingpeng 307
 Sato, Masayuki 378
 Shang, Jiandong 390
 Shao, Wenpei 282
 Sharma, Yanshul 405
 Sheng, Quan Z. 47, 174
 Song, Mingyang 307
 Sun, Caihua 140
 Sun, Siqing 331
 Sun, Wei 140
 Tan, Jingjing 58
 Tan, Sixiang 93, 128
 Tan, Wee Lum 236
 Tao, Ran 282
 Tran, Dai Hoang 47, 174
 Wan, Pengyu 282
 Wang, Fei 140
 Wang, Jack 34
 Wang, LieJun 128
 Wang, Rui 68
 Wang, Xiaofeng 187
 Wang, Xiao-Feng 200, 212
 Wang, Yingjing 343
 Wang, Yunlan 343
 Wang, Zhenyu 282
 Wei, Wenyu 128
 Wen, Jiebin 128
 Wu, Di 224, 270
 Wu, Jigang 1
 Wu, Run 224, 270
 Wu, Yalan 1
 Wu, Yongzhi 128
 Wu, Yu 140
 Wu, Zhize 187
 Wu, Zhi-Ze 200, 212
 Wu, Zi-Jun 200, 212
 Xu, Li-Xiang 212
 Yan, Dandan 365
 Yang, Kai 68
 Yang, Wenzhong 93, 128
 Yu, Weijie 93
 Zaib, Munazza 47
 Zeng, Yuying 307
 Zhang, Dongmei 58
 Zhang, Hongyang 58
 Zhang, Hui 331
 Zhang, Lianyi 68
 Zhang, Songyang 140
 Zhang, Wei E. 47
 Zhang, Wei Emma 174
 Zhang, Yong 355
 Zhang, Yujie 247, 259
 Zhang, Zhenning 58
 Zhao, Shipeng 318
 Zheng, Lulu 1
 Zheng, Qilong 22
 Zhou, Gangqiang 224
 Zhou, Yang 81
 Zhu, Lingxiao 104
 Zhu, Ming 282
 Zou, Le 187, 200, 212