applied sciences

Special Issue Reprint

Collaborative Learning and Optimization Theory and Its Applications

Edited by
Yue Wu, Mingyang Zhang, Qiguang Miao and Kai Qin

mdpi.com/journal/applsci

MDPI

Collaborative Learning and Optimization Theory and Its Applications

Collaborative Learning and Optimization Theory and Its Applications

Guest Editors

Yue Wu
Mingyang Zhang
Qiguang Miao
Kai Qin

MDPI

Basel • Beijing • Wuhan • Barcelona • Belgrade • Novi Sad • Cluj • Manchester

Guest Editors

Yue Wu
Department of Computer Science and Technology
Xidian University
Xi'an
China

Mingyang Zhang
School of Electronic Engineering
Xidian University
Xi'an
China

Qiguang Miao
School of Computer Science and Technology
Xidian University
Xi'an
China

Kai Qin
Department of Computer Science and Software Engineering
Swinburne University of Technology
Hawthorn
Australia

Editorial Office
MDPI AG
Grosspeteranlage 5
4052 Basel, Switzerland

This is a reprint of the Special Issue, published open access by the journal *Applied Sciences* (ISSN 2076-3417), freely accessible at: https://www.mdpi.com/journal/applsci/special_issues/88G3Z8K1I4.

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

Lastname, A.A.; Lastname, B.B. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-7258-9123-8 (Hbk)
ISBN 978-3-7258-9124-5 (PDF)
https://doi.org/10.3390/books978-3-7258-9124-5

© 2026 by the authors. Articles in this reprint are Open Access and distributed under the Creative Commons Attribution (CC BY) license. The reprint as a whole is distributed by MDPI under the terms and conditions of the Creative Commons Attribution-NonCommercial-NoDerivs (CC BY-NC-ND) license (https://creativecommons.org/licenses/by-nc-nd/4.0/).

Contents

Yue Wu, Mingyang Zhang, Qiguang Miao and Kai Qin
Collaborative Learning and Optimization Theory and Its Applications
Reprinted from: *Appl. Sci.* **2026**, *16*, 6877, https://doi.org/10.3390/app16146877 . . . . . **1**

Zhiyuan Li, Chenhui Ju, Hao Li and Maoguo Gong
Multi-Objective Federated Learning via Evolutionary Knowledge Transfer
Reprinted from: *Appl. Sci.* **2026**, *16*, 5094, https://doi.org/10.3390/app16105094 . . . . . **4**

Tao Wu, Shuo Xiong, Qiuhang Chen, Huaizheng Liu, Weijun Cao and Haoran Tuo
SupGAN: A General Super-Resolution GAN-Promoting Training Method
Reprinted from: *Appl. Sci.* **2025**, *15*, 9231, https://doi.org/10.3390/app15179231 . . . . . **27**

Hang Zhou, Tianning Shang, Yongchuan Wang and Long Zuo
Salp Swarm Algorithm Optimized A* Algorithm and Improved B-Spline Interpolation in Path Planning
Reprinted from: *Appl. Sci.* **2025**, *15*, 5583, https://doi.org/10.3390/app15105583 . . . . . **41**

Zhengyang Cao and Gang Chen
Advanced Cooperative Formation Control in Variable-Sweep Wing UAVs via the MADDPG–VSC Algorithm
Reprinted from: *Appl. Sci.* **2024**, *14*, 9048, https://doi.org/10.3390/app14199048 . . . . . **67**

Boliang Sun and Chao Chen
Sequence-Information Recognition Method Based on Integrated mDTW
Reprinted from: *Appl. Sci.* **2024**, *14*, 8716, https://doi.org/10.3390/app14198716 . . . . . **99**

Nataliya Shakhovska, Oleh Zherebetskyi and Serhii Lupenko
Model for Determining the Psycho-Emotional State of a Person Based on Multimodal Data Analysis
Reprinted from: *Appl. Sci.* **2024**, *14*, 1920, https://doi.org/10.3390/app14051920 . . . . . **119**

Baobao Liu, Heying Wang, Zifan Cao, Yu Wang, Lu Tao, Jingjing Yang and Kaibing Zhang
PRC-Light YOLO: An Efficient Lightweight Model for Fabric Defect Detection
Reprinted from: *Appl. Sci.* **2024**, *14*, 938, https://doi.org/10.3390/app14020938 . . . . . **143**

Fanxiao Yi, Weishi Li, Mengjie Huang, Yingchang Du and Lei Ye
A High-Quality Hybrid Mapping Model Based on Averaging Dense Sampling Parameters
Reprinted from: *Appl. Sci.* **2024**, *14*, 335, https://doi.org/10.3390/app14010335 . . . . . **164**

applied sciences

MDPI

Editorial

Collaborative Learning and Optimization Theory and Its Applications

Yue Wu [1,*], Mingyang Zhang [2], Qiguang Miao [1] and Kai Qin [3]

[1] Key Laboratory of Collaborative Intelligence Systems, Ministry of Education, Department of Computer Science and Technology, Xidian University, Xi'an 710071, China; qgmiao@xidian.edu.cn
[2] Key Laboratory of Collaborative Intelligence Systems, Ministry of Education, School of Electronic Engineering, Xidian University, Xi'an 710071, China; myzhang@xidian.edu.cn
[3] School of Science, Computing and Emerging Technologies, Swinburne University of Technology, Hawthorn, VIC 3122, Australia; kqin@swin.edu.au
* Correspondence: ywu@xidian.edu.cn

1. Introduction

Collaborative learning and optimization theory have become increasingly important paradigms for addressing complex intelligent system problems in modern machine learning and computational intelligence [1–4]. By enabling multiple models, agents, or computational components to jointly learn and optimize, these approaches provide strong capabilities for handling high-dimensional data, dynamic environments, and distributed decision-making problems [5,6]. With the rapid development of intelligent systems [7–10], the aim of this Special Issue was to collect recent advances in collaborative learning and optimization theory and its applications across diverse real-world scenarios.

2. Recent Advances

Recent studies have shown that collaborative learning and optimization have been widely applied in intelligent system scenarios such as collaborative computing environments, traffic forecasting and routing optimization, traffic signal control, person re-identification, large-scale e-learning, and privacy-preserving federated learning [11–13]. These developments indicate that collaborative learning and optimization are not limited to a single algorithmic framework, but provide general methodological support for perception, decision-making, coordination, and distributed intelligence [14–16]. Against this background, the contributions published in this Special Issue, "Collaborative Learning and Optimization Theory and Its Applications," further demonstrate the practical value of these methods in diverse application domains.

The contributions published in this Special Issue, "Collaborative Learning and Optimization Theory and Its Applications," reflect the latest progress in collaborative intelligence, optimization methodologies, and their applications in perception, control, sequence modeling, and distributed learning systems.

One group of works focus on collaborative perception and intelligent modeling in complex environments. These studies explore how optimized learning strategies and efficient representation models can improve system performance in tasks such as dense mapping, industrial inspection, and visual understanding. Hybrid modeling techniques are introduced to enhance stability and accuracy in dense sampling-based mapping scenarios (contribution 1), while lightweight learning architectures are proposed to achieve efficient and accurate defect detection in industrial environments (contribution 2). In addition, multimodal learning strategies further extend collaborative modeling capabilities by

integrating heterogeneous data sources for human state understanding (contribution 3), and generative learning methods are explored to improve super-resolution reconstruction quality and training robustness (contribution 7).

Another important direction is sequence modeling and structured data analysis under collaborative optimization frameworks. Improved temporal alignment methods are proposed to enhance sequence recognition performance by integrating multiple dynamic time warping strategies (contribution 4), demonstrating the effectiveness of collaborative feature alignment in structured temporal learning tasks.

In the domain of collaborative control and optimization, several studies investigate multi-agent coordination and intelligent decision-making strategies. Reinforcement learning-based formation control methods have been developed for unmanned aerial vehicle systems, enabling more stable and adaptive cooperative behaviors in dynamic environments (contribution 5). Furthermore, hybrid optimization strategies combining swarm intelligence, heuristic search, and interpolation techniques are proposed for path planning problems, significantly improving trajectory optimization and computational efficiency (contribution 6).

Finally, distributed and privacy-preserving collaborative learning has also received increasing attention. A multi-objective federated learning framework based on evolutionary knowledge transfer is introduced to enhance optimization performance in decentralized learning environments (contribution 8), demonstrating the effectiveness of collaborative optimization under data privacy constraints.

3. Future Outlook

The papers included in this Special Issue collectively demonstrate the broad applicability and effectiveness of collaborative learning and optimization theory in addressing complex intelligent system problems. From perception and sequence modeling to control, optimization, and federated learning, these contributions highlight the importance of collaboration mechanisms in improving learning efficiency, robustness, and scalability. Future research is expected to further explore more unified collaborative frameworks that integrate learning, optimization, and decision-making under increasingly complex and distributed environments.

Funding: This research received no external funding.

Acknowledgments: Thanks are due to all the authors and peer reviewers for their valuable contributions to this Special Issue, as well as to the reviewers and editors for their valuable comments and feedback to help the authors improve their papers. Herewith, we congratulate all of the authors for their outstanding achievements on relevant topics.

Conflicts of Interest: The authors declare no conflicts of interest.

List of Contributions:

1. Yi, F.; Li, W.; Huang, M.; Du, Y.; Ye, L. A High-Quality Hybrid Mapping Model Based on Averaging Dense Sampling Parameters. *Appl. Sci.* **2024**, *14*, 335. https://doi.org/10.3390/app14010335.
2. Liu, B.; Wang, H.; Cao, Z.; Wang, Y.; Tao, L.; Yang, J.; Zhang, K. PRC-Light YOLO: An Efficient Lightweight Model for Fabric Defect Detection. *Appl. Sci.* **2024**, *14*, 938. https://doi.org/10.3390/app14020938.
3. Shakhovska, N.; Zherebetskyi, O.; Lupenko, S. Model for Determining the Psycho-Emotional State of a Person Based on Multimodal Data Analysis. *Appl. Sci.* **2024**, *14*, 1920. https://doi.org/10.3390/app14051920.
4. Sun, B.; Chen, C. Sequence-Information Recognition Method Based on Integrated mDTW. *Appl. Sci.* **2024**, *14*, 8716. https://doi.org/10.3390/app14198716.

5. Cao, Z.; Chen, G. Advanced Cooperative Formation Control in Variable-Sweep Wing UAVs via the MADDPG–VSC Algorithm. *Appl. Sci.* **2024**, *14*, 9048. https://doi.org/10.3390/app14199048.
6. Zhou, H.; Shang, T.; Wang, Y.; Zuo, L. Salp Swarm Algorithm Optimized A* Algorithm and Improved B-Spline Interpolation in Path Planning. *Appl. Sci.* **2025**, *15*, 5583. https://doi.org/10.3390/app15105583.
7. Wu, T.; Xiong, S.; Chen, Q.; Liu, H.; Cao, W.; Tuo, H. SupGAN: A General Super-Resolution GAN-Promoting Training Method. *Appl. Sci.* **2025**, *15*, 9231. https://doi.org/10.3390/app15179231.
8. Li, Z.; Ju, C.; Li, H.; Gong, M. Multi-Objective Federated Learning via Evolutionary Knowledge Transfer. *Appl. Sci.* **2026**, *16*, 5094. https://doi.org/10.3390/app16105094.

References

1. Song, G.; Chai, W. Collaborative learning for deep neural networks. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: Montréal, Canada, 2018; Volume 31, pp. 1832–1841.
2. Dorner, F.E.; Konstantinov, N.; Pashaliev, G.; Vechev, M. Incentivizing honesty among competitors in collaborative learning and optimization. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: New Orleans, LA, USA, 2023; Volume 36, pp. 7659–7696.
3. Wu, Y.; Ding, H.; Gong, M.; Qin, A.K.; Ma, W.; Miao, Q.; Tan, K.C. Evolutionary multiform optimization with two-stage bidirectional knowledge transfer strategy for point cloud registration. *IEEE Trans. Evol. Comput.* **2024**, *28*, 62–76. [CrossRef]
4. Gong, M.; He, Y.; Li, H.; Wu, Y.; Zhang, M.; Wang, S.; Luo, T. Frontiers of collaborative intelligence systems. *J. Inf. Intell.* **2024**, *2*, 14–27. [CrossRef]
5. Nedic, A.; Ozdaglar, A. Distributed subgradient methods for multi-agent optimization. *IEEE Trans. Autom. Control* **2009**, *54*, 48–61. [CrossRef]
6. Cemri, M.; Pan, M.Z.; Yang, S.; Agrawal, L.A.; Chopra, B.; Tiwari, R.; Keutzer, K.; Parameswaran, A.; Klein, D.; Ramchandran, K.; et al. Why do multi-agent LLM systems fail? In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc.: San Diego, CA, USA, 2025; Volume 38.
7. Kang, Y.; Kim, J. ChatMOF: An artificial intelligence system for predicting and generating metal-organic frameworks using large language models. *Nat. Commun.* **2024**, *15*, 4705. [CrossRef] [PubMed]
8. Sheth, A.; Roy, K.; Gaur, M. Neurosymbolic artificial intelligence (why, what, and how). *IEEE Intell. Syst.* **2023**, *38*, 56–62. [CrossRef]
9. Zhang, Z.; Wen, F.; Sun, Z.; Guo, X.; He, T.; Lee, C. Artificial intelligence-enabled sensing technologies in the 5G/internet of things era: From virtual reality/augmented reality to the digital twin. *Adv. Intell. Syst.* **2022**, *4*, 2100228. [CrossRef]
10. Cao, L. Decentralized ai: Edge intelligence and smart blockchain, metaverse, web3, and desci. *IEEE Intell. Syst.* **2022**, *37*, 6–19. [CrossRef]
11. Derakhshan, B.; Mahdiraji, A.R.; Abedjan, Z.; Rabl, T.; Markl, V. Optimizing machine learning workloads in collaborative environments. In Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, Portland, OR, USA, 14–19 June 2020; pp. 1701–1716.
12. Guo, Z.; Zhang, Y.; Lv, J.; Liu, Y.; Liu, Y. An online learning collaborative method for traffic forecasting and routing optimization. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 6634–6645. [CrossRef]
13. Zhang, Y.; Liu, Y.; Gong, P.; Li, P.; Fan, M.; Sartoretti, G. Unicorn: A Universal and Collaborative Reinforcement Learning Approach Toward Generalizable Network-Wide Traffic Signal Control. *IEEE Trans. Intell. Transp. Syst.* **2026**, *27*, 4552–4568. [CrossRef]
14. Gao, Z.; Wei, S.; Guan, W.; Zhu, L.; Wang, M.; Chen, S. Identity-guided collaborative learning for cloth-changing person reidentification. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 2819–2837. [CrossRef] [PubMed]
15. Ma, H.; Fu, X.; Tang, W.; Chen, M.; Zhu, H.; Li, K. Three-Stage Grouping Optimization for Large-Scale Collaborative E-Learning via Knowledge Graph and E-CARGO. *IEEE Trans. Syst. Man. Cybern. Syst.* **2026**, *56*, 2684–2698. [CrossRef]
16. Gu, B.; Xu, A.; Huo, Z.; Deng, C.; Huang, H. Privacy-preserving asynchronous vertical federated learning algorithms for multiparty collaborative learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 6103–6115. [CrossRef] [PubMed]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

Multi-Objective Federated Learning via Evolutionary Knowledge Transfer

Zhiyuan Li [1], Chenhui Ju [1], Hao Li [1] and Maoguo Gong [1,2,*]

[1] School of Electronic Engineering, Xidian University, Xi'an 710071, China; zhiyuanjli@stu.xidian.edu.cn (Z.L.); 25241215060@stu.xidian.edu.cn (C.J.); haoli@xidian.edu.cn (H.L.)
[2] College of Mathematics Science, Inner Mongolia Normal University, Hohhot 010022, China
* Correspondence: mggong@mail.xidian.edu.cn

Abstract

Federated learning enables multiple clients to collaboratively train a global model without exchanging raw data, thereby alleviating data silos and privacy concerns. However, federated learning model design is still challenged by complex architecture and hyperparameter optimization, especially under heterogeneous data distributions. To address these issues, this paper proposes a multi-objective federated learning framework based on particle swarm optimization (PSO). Specifically, neural architecture search and training-parameter optimization are formulated as a multi-objective evolutionary optimization problem, where predictive performance and model complexity are optimized jointly. Furthermore, to improve robustness under non-independent and identically distributed (non-IID) data, a clustered multi-population extension is developed. In this framework, each cluster is associated with a dedicated particle population. An inter-population evolutionary knowledge transfer mechanism is then introduced to enable effective sharing of search experience across clusters, thereby improving the optimization efficiency and adaptability of federated learning under heterogeneous environments. Experiments on MNIST and CIFAR10 under both IID and non-IID settings demonstrate that the proposed methods achieve a superior trade-off between predictive accuracy and model complexity and outperform baseline approaches in robustness, search efficiency, and adaptability to imbalanced distributed environments.

Keywords: federated learning; particle swarm optimization; evolutionary knowledge transfer; multi-objective optimization; neural architecture search

1. Introduction

Federated learning (FL) has emerged as an effective distributed learning paradigm that enables multiple clients to collaboratively train a global model without sharing raw data [1]. By keeping private data on local devices and exchanging only model parameters or updates, FL provides a practical solution to data silos and privacy concerns in many real-world applications. With smartphones and other edge devices becoming dominant carriers of user data, data processing has gradually shifted from traditional centralized platforms to distributed and mobile environments [2]. However, despite its advantages in privacy preservation, FL still faces substantial challenges in model design and optimization, especially when client data exhibit heterogeneous and non-independent identically distributed (non-IID) characteristics [3].

In practical FL systems, the overall learning performance is strongly influenced by both the neural network architecture and training hyperparameters. On the one hand,

model structure directly affects predictive accuracy and model complexity. On the other hand, training-related parameters, such as learning rate, local update frequency, and communication rounds, also affect communication efficiency and convergence behavior. As a result, the design of FL models is essentially a coupled optimization problem involving multiple conflicting objectives. Traditional manual tuning and grid search methods are often computationally expensive and difficult to scale to high-dimensional search spaces, particularly when both architecture design and federated training parameters need to be optimized jointly. Moreover, in realistic deployment scenarios, it is also necessary to design effective strategies for model allocation and optimization under heterogeneous client environments [4].

Evolutionary optimization provides a promising solution to such problems. Owing to its gradient-free nature and strong global search capability, it has been widely used in black-box optimization, neural architecture search, and complex multimodal optimization problems [5,6]. In particular, particle swarm optimization (PSO) is attractive because of its simple implementation, efficient population-based search mechanism, and suitability for large search spaces [7,8]. These characteristics make PSO a natural candidate for federated learning optimization, where the search space often includes both discrete architectural variables and continuous training hyperparameters. Compared with conventional grid search, PSO-based update strategies can substantially reduce computational cost while maintaining efficient exploration ability in large search spaces [9,10]. However, directly applying a single-population optimization strategy to FL may still be insufficient in heterogeneous environments because client distributions can vary significantly and may induce highly inconsistent optimization landscapes.

To better address this issue, it is necessary to exploit the structural relationships among different client groups. In heterogeneous FL scenarios, different clusters of clients may accumulate useful search experience under their own data distributions. If such experience can be shared effectively, the optimization process may avoid redundant exploration, accelerate convergence, and improve generalization across clusters. This motivates the introduction of evolutionary knowledge transfer into federated learning optimization. Rather than treating all clients as belonging to a single homogeneous population, a clustered multi-population framework makes it possible to preserve cluster-specific search characteristics while enabling information exchange across related populations.

Based on this observation, this paper proposes a multi-objective particle swarm optimization framework for federated learning. The base framework formulates federated model design as a multi-objective optimization problem in which predictive performance and model complexity are optimized simultaneously. Furthermore, to cope with data heterogeneity, a clustered multi-population extension is developed. In this extension, clients are dynamically grouped according to their data characteristics, and each cluster is assigned to a specialized particle population. More importantly, an inter-population evolutionary knowledge transfer mechanism is introduced, through which cluster-specific nondominated solutions can provide transferable search experience for other populations. In this way, the proposed method not only supports efficient federated architecture optimization but also improves robustness and adaptability under non-IID conditions.

The main contributions of this paper are summarized as follows:

- We propose a multi-objective PSO-based federated learning framework that jointly optimizes predictive performance and model complexity in neural architecture and training-parameter search.
- We develop a clustered multi-population extension for heterogeneous federated learning in which an inter-population evolutionary knowledge transfer mechanism is designed to improve search efficiency and robustness under non-IID data distributions.

https://doi.org/10.3390/app16105094

- We conduct comprehensive experiments on MNIST and CIFAR10 under both IID and non-IID settings, and the results demonstrate that the proposed methods achieve better trade-offs between accuracy and complexity than the baseline methods.

2. Background

In this section, we first introduce federated learning and clustered federated learning, where the latter further considers client heterogeneity in distributed learning. In addition, we review particle swarm optimization, which is closely related to the proposed optimization framework.

2.1. Federated Learning

One of the primary focuses of FL is protecting the data privacy and security of devices. However, this process differs from traditional privacy-preserving methods, such as differential privacy [11] and secure multi-party computation [12]. Most application scenarios for the aforementioned methods involve the encryption, transmission, and decryption of raw data, whereas the encrypted objects in FL scenarios are the trained models and gradients. In actual production scenarios, the encryption process in FL is often combined with technologies such as differential privacy and homomorphic encryption to mitigate the untrusted server problem and defend against inversion attacks from malicious Byzantine users [13,14]. Another important characteristic of FL is that it is a distributed learning technology with privacy-preserving attributes [15,16]. Traditional distributed learning enjoys obvious advantages over centralized learning when data volume and computational resources are sufficient [17]. This method deploys the original independent and identically distributed data and models onto edge nodes, and the central server provides unified scheduling so that all nodes perform model training simultaneously. However, in the FL scenario, data privacy is stringently defined, which means that the transmission and communication of raw data cannot be performed during any process. This inevitably causes the data across various edge devices to be non-independent and identically distributed, which undoubtedly brings new challenges to the training and optimization of artificial intelligence models [18]. Although federated learning still faces some unresolved issues in the current environment, it has played an important role during its development in already-deployed sectors, such as the medical industry, the financial industry, and terminal service providers [19]. Meanwhile, in face detection tasks within the security domain [20], the capacity of federated learning to enable multiple parties to collaboratively build safe, efficient, and compliant distributed models can help institutions to achieve cross-platform knowledge sharing [21,22].

Building upon previous research, more advanced methods have been proposed. Personalized federated averaging was proposed to address the issue of model personalization by decoupling the model from the global model [23,24]. Since the local models can be iteratively trained, the global model achieves better performance by utilizing higher-quality local updates. The quantized networks on client devices are optimized to reduce weight differences through a self-learning quantization factor method, which simultaneously reduces both uplink and downlink communication costs [25]. Additionally, an approach to mitigate the challenge of limited and poor-quality data within data silos is proposed [26].

2.1.1. Federated Averaging

Google proposed a horizontal federated learning solution for Android devices in 2017, known as federated averaging (FedAvg) [27]. In this framework, users update model parameters locally and upload them to the cloud so that multiple data owners can jointly train a global model. A secure aggregation mechanism is also introduced to protect user

https://doi.org/10.3390/app16105094

privacy during parameter uploading. Unlike a centralized neural network, this method optimizes the objective function in a distributed manner.

Let C denote the number of independent clients, let P_c denote the sample index set of client c, let $p_c = |P_c|$ be the number of samples on client c, and let $p = \sum_{c=1}^{C} p_c$ be the total number of samples across all clients. The global objective can be written as

$$\min_{w} l(w) = \sum_{c=1}^{C} \frac{p_c}{p} L_c(w), \tag{1}$$

where the local objective of client c is as follows:

$$L_c(w) = \frac{1}{p_c} \sum_{i \in P_c} l_i(w), \tag{2}$$

Here, $l(w)$ denotes the global objective, $l_i(w) = f(x_i, y_i; w)$ denotes the loss on sample (x_i, y_i), and w denotes model parameters.

FedAvg is effective in reducing communication cost. By extending federated stochastic gradient descent [28], it increases local training epochs and reduces the need for frequent communication. Specifically, FedAvg can be described in three steps:

First, each client downloads the global model from the server and updates the local model by performing stochastic gradient descent (SGD) [29] on its local dataset. In communication round t, the model of the c-th client (where $c \in C$) is defined as w_c^t, and the update amount can be expressed as

$$\Delta w_c^{t+1} = f(w^t, D_c) - w^t, \quad c = 1, \ldots, C, \tag{3}$$

where $f(\cdot)$ denotes SGD, and D_c denotes the local data of client c.

Second, each client uploads its trained model to the central server, and these parameters are aggregated by an averaging method. The formula can be defined as

$$w^{t+1} = w^t + \sum_{c=1}^{C} \frac{D_c}{D} \Delta w_c^{t+1}. \tag{4}$$

where w_t denotes the global model at round t.

Third, the server broadcasts the aggregated model to all clients:

$$w_c^{t+1,0} = w^{t+1}, \quad c = 1, \ldots, C. \tag{5}$$

Overall, FedAvg adopts a training mechanism of "local multi-step updates and global weighted aggregation". This method reduces communication frequency by executing multiple rounds of local optimization on the client side. Meanwhile, it utilizes a weighted-averaging strategy on the server side to fuse the training results of different clients, thereby achieving continuous optimization of the global model while preserving data locality. After multiple iterations, the model parameters gradually approach the optimal solution, ultimately yielding a global model that simultaneously accommodates the data-distribution characteristics of individual clients.

2.1.2. Clustered Federated Learning

In federated learning, because data sources and computing nodes both come from end-user personal devices, it is therefore unavoidable that data heterogeneity problems arise, namely the problem of non-independent and identically distributed data. In applications such as recommender systems and personalized advertising [30,31], fully utilizing data

https://doi.org/10.3390/app16105094

heterogeneity is particularly important. This may not only improve user experience but also help enterprises to obtain better application effects and benefits. The above situations indicate that fully utilizing the data heterogeneity among users is of great significance. On the one hand, the data owned by a single device are usually limited, and it is therefore necessary to effectively mine the similarity among users so as to improve model performance. On the other hand, if the data of all users are simply regarded as IID samples, it is difficult to realize personalized prediction, thereby limiting the performance of the model in practical applications [32–34]. Because the non-IID problem is widespread in federated learning scenarios, it is crucial to build a more reasonable model to deal with various data distributions [35]. Traditional methods have been shown to be qualified under some forms of non-IID data, but, when facing highly skewed data distributions, they suffer from significant performance degradation [36]. To address this problem, FedProx [37] provides a solution to the non-IID problem by using convergence guarantees and constraint strategies. However, clustered federated learning [38] reassigns clients to the most suitable clusters by evaluating the performance of client models.

In a typical clustered federated learning setting, we assume there exist k distinct data distributions, $\mathcal{D}_1, \ldots, \mathcal{D}_k$, and that m clients are partitioned into k pairwise disjoint clusters, $S_1^*, \ldots, S_k^*$. We assume each client $i \in S_j^*$ belonging to S_j^* contains n independent and identically distributed data points $z^{i,1}, \ldots, z^{i,n}$, which are sampled from the distribution $\mathcal{D}_j$. Among them, each data point $z^{i,\ell}$ consists of a pair of variables representing a feature and a response, denoted as $z^{i,\ell} = (x^{i,\ell}, y^{i,\ell})$.

Let $f(\theta; z) : \Theta \to \mathbb{R}$ denote the loss function associated with the data point z, where $\Theta \subseteq \mathbb{R}^d$ is the parameter space. In this paper, we take $\Theta = \mathbb{R}^d$. The overall loss function to minimize is as follows:

$$F^j(\theta) := \mathbb{E}_{z \sim \mathcal{D}_j}[f(\theta; z)], \quad j \in [k]. \tag{6}$$

In our problem, the data distributions across all user devices possess a cluster structure. A natural idea is to adopt off-the-shelf clustering algorithms (such as k-means) to estimate the cluster assignments [38]. Because the data are stored on individual worker machines, taking into account the massive communication overhead, it is infeasible to upload all data to a central machine and cluster the raw data points. Therefore, a straightforward approach is to let each client train a local model $\widetilde{\theta}_i$ using its local data and send it to the server. Subsequently, the server estimates the cluster assignment of each client by running k-means on these $\widetilde{\theta}_i$ models. After obtaining the classification estimates, the central server can independently run any federated learning aggregation algorithm within each cluster.

Algorithm 1 is a modular algorithm. It has flexibility in the selection of the clustering subroutine and the optimization subroutine, and these subroutines can be adjusted according to specific problem instances.

Algorithm 1 One-Shot Clustering Algorithm for Federated Learning

1: Clients send the solutions derived by minimizing the empirical risk

$$\widetilde{\theta}_i := \arg\min_{\theta \in \Theta} F_i(\theta), \quad \forall\, i \in [m]$$

to the server.
2: The server clusters $\{\widetilde{\theta}_i\}_{i=1}^m$ into $S_1, \ldots, S_k$.
3: Within each cluster S_j, $j \in [k]$, run a federated learning algorithm to obtain the final solution $\hat{\theta}_j$.

The clustered federated learning framework introduces a client-clustering mechanism on the basis of the traditional federated learning framework in order to address the problem

https://doi.org/10.3390/app16105094

of heterogeneous non-IID data distributions in federated learning. Instead of maintaining a single global model, the server clusters clients according to their models or data characteristics to group those with similar data distributions. During each training round, after each client completes model updating based on local data, the model parameters are uploaded to the server. The server performs clustered federated aggregation (C-FedAvg) by averaging parameters only among client models within the same cluster to obtain the corresponding cluster-level global model. This method can reduce interference among different data distributions so that the model can better adapt to various data characteristics. By introducing the client-clustering mechanism, clustered federated learning aims to maintain the privacy-preserving properties of federated learning while improving the training stability and overall performance of the model under non-independent and identically distributed data environments.

2.2. Particle Swarm Optimization

Particle swarm optimization (PSO) [39] is an intelligent optimization algorithm based on populations. Its inspiration comes from the foraging behavior of bird flocks. Because PSO has the advantages of conceptual simplicity and ease of implementation, it has been widely applied in many fields in recent years and has achieved good results. A particle has only two attributes, namely velocity and position. Velocity represents the speed of movement, while position represents the direction of movement. Each particle independently identifies and records its personal best in the search space, which is then shared with all other particles in the swarm. The best personal best discovered across the swarm is adopted as the current global best. Subsequently, all particles adjust their velocities and positions based on both their individual best and the shared global best.

In PSO, the position of particle i is written as $\mathbf{x}^i = (x^{i1}, x^{i2}, \ldots, x^{id})$, where d is the dimensionality of the search space. The velocity of particle i is denoted by

$$\mathbf{v}^i = (v^{i1}, v^{i2}, \ldots, v^{id}). \tag{7}$$

During initialization, both positions and velocities are randomly generated. The iterative update stage then follows:

$$\mathbf{v}^{id}(k+1) = w\,\mathbf{v}^{id}(k) + c_1 r_1\big(\mathbf{p}^{id} - \mathbf{x}^{id}(k)\big) + c_2 r_2\big(\mathbf{p}^{gd} - \mathbf{x}^{id}(k)\big), \tag{8}$$

$$\mathbf{x}^{id}(k+1) = \mathbf{x}^{id}(k) + \mathbf{v}^{id}(k+1), \tag{9}$$

where w is the inertia weight. It controls the influence of the current velocity on the next update, thereby balancing exploration and exploitation. If $\omega = 0$, the velocity has no memory because it depends only on the current and historical best positions. Suppose that one particle is already located at the global best position. While this particle remains stationary, the others will fly toward the weighted center of their personal and global bests, causing the entire swarm to shrink toward the current global optimum. Here, $\mathbf{x}^{id}(k)$ and $\mathbf{v}^{id}(k)$ denote the position and velocity of particle i in dimension d at iteration k, respectively; $\mathbf{p}^{id}$ denotes the component of the historical best position of particle i (personal best, *pbest*) in dimension d; $\mathbf{p}^{gd}$ denotes the component of the global best position of the whole population (global best, *gbest*) in dimension d. c_1 and c_2 are learning factors, which control the extent to which the particle moves toward the personal best and the global best. Lower values allow particles to wander outside the target region before being pulled back, while higher values cause particles to suddenly rush toward or overshoot the target region. $r_1, r_2 \sim U(0,1)$ are random numbers on the interval $[0,1]$ and are used to enhance the randomness and diversity of the algorithm.

PSO is a population-based optimization algorithm in which particles cooperatively search the solution space through information sharing. In the particle swarm algorithm,

https://doi.org/10.3390/app16105094

intelligent agents cooperate to adapt to the environment while demonstrating a clear ability to interact with it.

Standard federated averaging methods provide the foundation for privacy-preserving distributed learning, but their performance often degrades under heterogeneous and non-IID client distributions. Clustered federated learning extends this paradigm by grouping clients with similar data characteristics, thereby improving adaptability in heterogeneous environments. However, both standard and clustered federated learning frameworks still rely heavily on manually selected model architectures and training hyperparameters. This motivates the introduction of PSO-based multi-objective optimization, which can automatically search for suitable model configurations while balancing predictive performance and model complexity. From this perspective, the proposed method can be viewed as a unified framework that combines clustered federated learning with PSO-based optimization to address both client heterogeneity and model-design complexity.

3. Methodology

This section presents the proposed PSO-based federated learning framework and its clustered extension. The former addresses multi-objective architecture and hyperparameter optimization, while the latter introduces inter-population evolutionary knowledge transfer to improve robustness under heterogeneous data distributions.

3.1. Multi-Objective Particle Swarm Optimization for Federated Learning

Figure 1 illustrates the overall framework of the proposed method. The server first initializes the global model and the PSO parameters, including particle positions, velocities, population size, inertia weight, and learning factors. The current global model is then distributed to all participating clients. Each client performs local training on its private dataset and uploads the updated model parameters to the server. After receiving local updates, the server aggregates them to obtain a new global model.

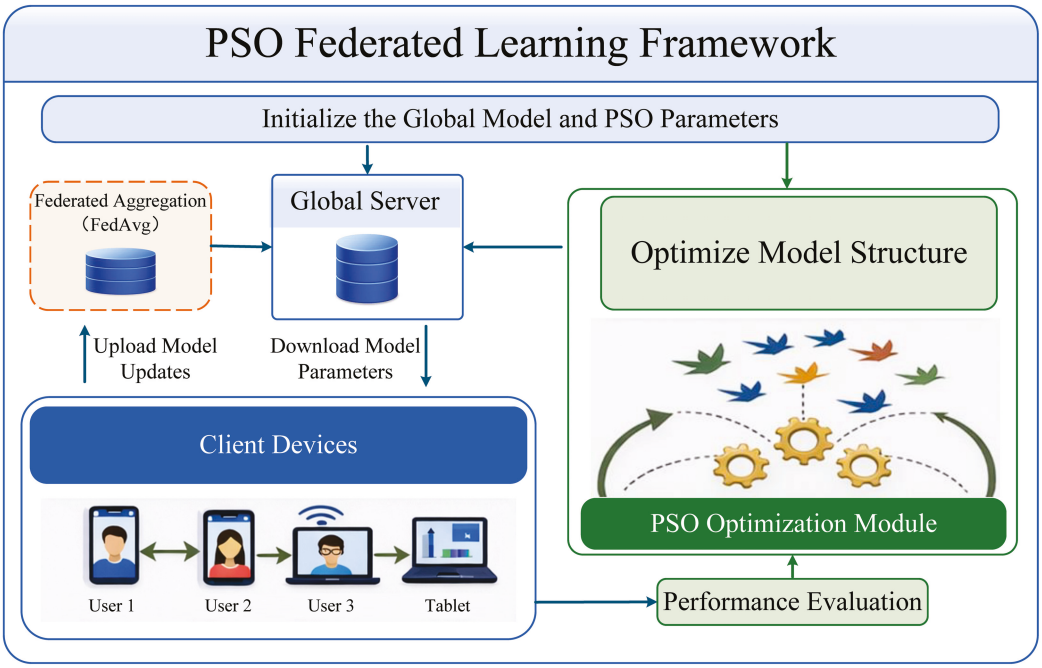


Figure 1. Framework of the multi-objective particle swarm federated learning method.

On top of the federated training procedure, PSO is employed to jointly optimize model architectures and training hyperparameters. Specifically, each particle encodes a candidate configuration consisting of structural parameters and optimization-related variables, such

as learning rate and local training epochs. The performance evaluation module then assesses the federated learning result associated with each particle and computes its fitness according to predictive performance and model complexity. Based on the resulting fitness values, the personal best and global best particles are updated, and the swarm continues to evolve toward more promising configurations.

3.1.1. Objective Function Definition

Most existing federated learning algorithms pay more attention to model performance. However, in practical applications, the communication efficiency of devices is also an important indicator. In most cases, improving model performance and minimizing communication efficiency are contradictory. Therefore, we regard this problem as a multi-objective problem and model it as a bi-objective optimization function. Here, model complexity is used as a proxy for communication overhead since larger models generally incur higher transmission cost in federated learning:

$$\min_{W}(f_1, f_2)^T = \begin{cases} f_1 : \text{model complexity} \\ f_2 : \text{loss function} \end{cases} \tag{10}$$

where $f_1 = \sum_h \|W^{(h)}\|_0$ and $f_2 = \mathbb{E}_{(x,y)\sim D}[F(x, y; W)]$. Here, f_1 denotes model complexity, and f_2 denotes the global loss function of the federated model. Here, W denotes the model parameters of the current neural network, and h denotes the number of hidden layers. The definition of f_2 is the same as that in basic federated learning, where D denotes the current dataset and (x, y) denotes one training sample.

3.1.2. Search Space Design

To obtain the optimal federated learning structure, the search space includes most hyperparameters. The encoding results are as follows:

$$X = (\eta, \mu, e, conv1, conv2, k1, k2, fc1, fc2), \tag{11}$$

Here, η and μ denote the learning rate and local update frequency, respectively, while e represents the number of local training epochs. For the convolutional layers, $conv1$ and $conv2$ signify the numbers of kernels, with $k1$ and $k2$ denoting their respective sizes. Finally, $fc1$ and $fc2$ specify the numbers of neurons in the fully connected layers.

Because the numbers of convolutional-layer channels, the convolution-kernel sizes, and the numbers of nodes in the fully connected layers are all integer variables, it is necessary to discretize the particle positions when actually constructing the network structure.

Figure 2 shows a convolutional neural network (CNN) structure to be optimized, and its training dataset is the MNIST handwritten digit dataset. The network input is a grayscale image of size 28×28. Image features are extracted layer by layer through multiple convolutional layers while pooling operations reduce dimensionality and computational complexity. After convolutional feature extraction, the network integrates high-dimensional features through fully connected layers and finally outputs a 10-dimensional vector to represent the classification probabilities of digits 0–9. The number of convolutional channels, the kernel sizes, and the number of neurons in the fully connected layers are all treated as optimization parameters. Based on these, the optimal network structure is automatically searched within the predefined search space by the optimization algorithm.

https://doi.org/10.3390/app16105094

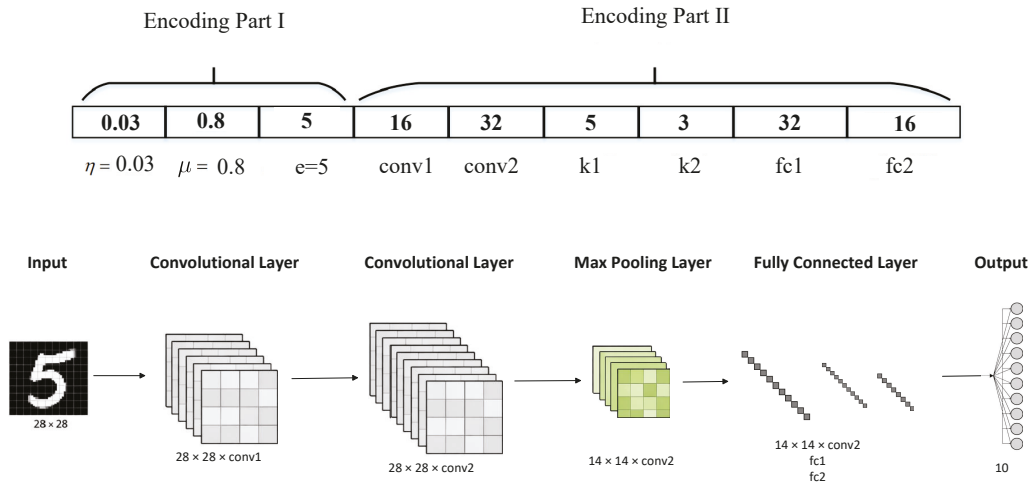


Figure 2. An example of the search space, including hyperparameters and model architecture parameters. Specifically, *conv*1 and *conv*2 represent 16 and 32 convolutional filter channels, respectively, and $fc1$ and $fc2$ represent 32 and 16 neurons in the fully connected layers, respectively. The padding of the convolutional layers is set to 'same', so the size of the feature maps output by each convolutional layer remains unchanged before the max pooling operation. Note that there must be at least one neuron in each hidden layer.

3.1.3. Multi-Objective PSO Federated Learning Optimization Procedure

In line 10 of Algorithm 2, in order to determine whether the current position is better than the historical best position, it is necessary to compare them by using the Pareto dominance relation. Specifically, the current position X_i^t must satisfy the following condition:

$$\forall k \in \{1, \ldots, K\}, \quad f_k(X_i^t) \leq f_k(P_i^t), \tag{12}$$

In this case, X_i^t Pareto dominates P_i^t ($X_i^t \prec P_i^t$) provided that $f_k(X_i^t) < f_k(P_i^t)$ holds for at least one $k \in 1, \ldots, K$. In this case, the current position is updated as the new personal best, namely $P_i^{t+1} = X_i^t$. If the historical best position dominates the current position ($P_i^t \prec X_i^t$), the personal best remains unchanged ($P_i^{t+1} = P_i^t$). Conversely, when X_i^t and P_i^t do not dominate each other, the new personal best is selected using a crowding-distance-based rule:

$$P_i^{t+1} = \begin{cases} X_i^t, & X_i^t \prec P_i^t, \\ P_i^t, & P_i^t \prec X_i^t, \\ \text{Select}(X_i^t, P_i^t), & \text{otherwise.} \end{cases} \tag{13}$$

In the nondominated case, the personal best is selected according to the crowding distance (CD), so the solution located in a less crowded region of the objective space is preferred. Specifically, for a nondominated set $\mathcal{F}$, the crowding distance of solution $z_j \in \mathcal{F}$ is computed as

$$CD(z_j) = \sum_{k=1}^{K} \frac{f_k(z_{j+1}) - f_k(z_{j-1})}{f_k^{\max} - f_k^{\min}}, \tag{14}$$

where $f_k^{\max}$ and $f_k^{\min}$ denote the maximum and minimum objective values of the k-th objective in $\mathcal{F}$, respectively, and z_{j-1} and z_{j+1} are the two neighboring solutions of z_j after sorting $\mathcal{F}$ according to the k-th objective. For boundary solutions, the crowding distance is

https://doi.org/10.3390/app16105094

set to $+\infty$. Therefore, when X_i^t and P_i^t do not dominate each other, the one with the larger crowding distance is selected:

$$\text{Select}(X_i^t, P_i^t) = \begin{cases} X_i^t, & CD(X_i^t) > CD(P_i^t), \\ P_i^t, & CD(P_i^t) > CD(X_i^t), \\ \text{random}(X_i^t, P_i^t), & CD(X_i^t) = CD(P_i^t). \end{cases} \tag{15}$$

Algorithm 2 Multi-Objective PSO Federated Learning

Input: particle number N, maximum iteration number $T_{\max}$, inertia weight ω, and learning factors c_1 and c_2
Output: Pareto nondominated solution set $A^{T_{\max}}$

Initialize particle positions X_i^0 and velocities V_i^0, where $i = 1, 2, \ldots, N$
Initialize personal best positions $P_i^0 \leftarrow X_i^0$
Initialize the external archive $A^0 \leftarrow \emptyset$
for $t = 0, 1, \ldots, T_{\max} - 1$ **do**
  for each particle $i = 1, 2, \ldots, N$ **do**
    Decode X_i^t into a federated parameter configuration $\Lambda_i^t = \phi(X_i^t)$
    Execute one round of federated training under Λ_i^t and obtain the global model w_i^{t+1}
    Compute the objective vector $\mathbf{F}(X_i^t) = \big(f_1(X_i^t), f_2(X_i^t)\big)$
    Update the personal best P_i^t according to the Pareto dominance relation
  end for
  Update the external archive:
    $A^{t+1} = \text{nondominated}\big(A^t \cup \{X_1^t, X_2^t, \ldots, X_N^t\}\big)$
  for each particle $i = 1, 2, \ldots, N$ **do**
    Select a leader G_i^{t+1} from A^{t+1}
    Update the velocity:
      $V_i^{t+1} = \omega V_i^t + c_1 r_1 (P_i^t - X_i^t) + c_2 r_2 (G_i^{t+1} - X_i^t)$
    Update the position: $X_i^{t+1} = X_i^t + V_i^{t+1}$
  end for
end for
Return $A^{T_{\max}}$

3.2. Clustered Multi-Objective Particle Swarm Optimization for Federated Learning

As shown in Figure 3, different from traditional federated learning, which only maintains a single global model, this framework simultaneously maintains multiple cluster-level models on the server side. First, the server initializes the global model and clusters clients according to client data characteristics or model-update information, thereby forming multiple user clusters. For each user cluster, the server separately maintains a corresponding cluster-level model and performs model aggregation within the cluster, thereby obtaining multiple server models that are updated in parallel. At the same time, the PSO optimization module is used to search the model-structure parameters and training hyperparameters in federated learning. Each particle represents one set of candidate parameter configurations for a certain cluster, and all particles for a given cluster are regarded as one population. Under the given parameter configuration, one round of clustered federated training is performed, and the corresponding fitness value is calculated through the performance-evaluation module. Subsequently, PSO updates the positions and velocities of the particles according to their personal and population best solutions. This iterative mechanism enables continuous exploration of promising parameter combinations in the search space. Throughout this iterative process, the framework maintains multiple cluster-level server models while utilizing inter-population genetic knowledge transfer to optimize both model

structures and training parameters. This approach realizes efficient federated model search and training under complex data-distribution environments.

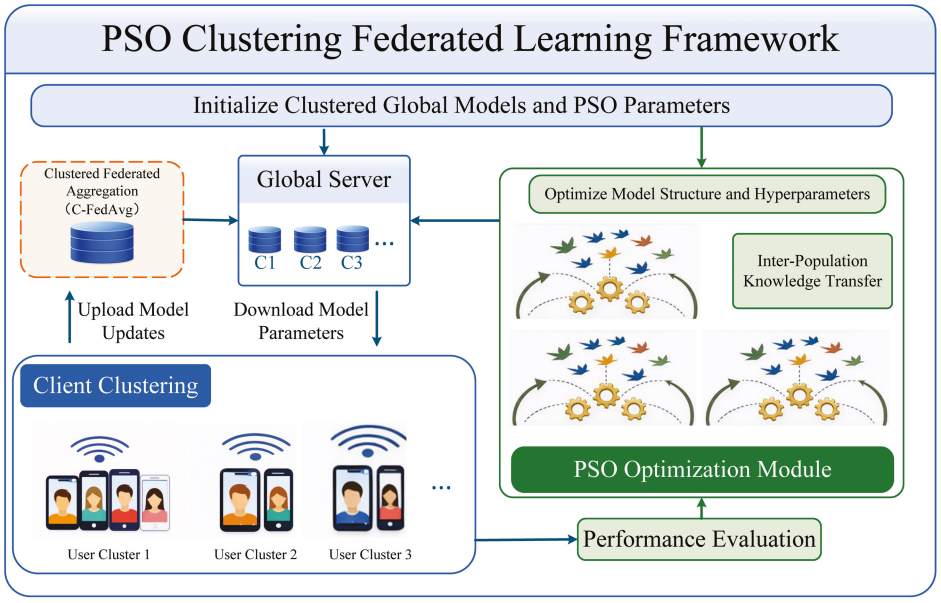


Figure 3. Framework of the clustered multi-objective particle swarm federated learning method.

In the proposed framework, each cluster-specific population evolves under a distinct local data distribution. The nondominated candidate solutions obtained in one population serve as transferable search experience for other populations through the joint evaluation and projection mechanism. This inter-population transfer helps to avoid redundant exploration and improves robustness under non-IID settings.

3.2.1. Objective Function Definition

Similar to the multi-objective federated learning method, we formulate the clustered optimization problem as

$$\min_{W}(f_1, f_2)^T = \begin{cases} f_1 : \text{model complexity} \\ f_2 : \text{clustered federated loss function}, \end{cases} \tag{16}$$

Here,

$$f_1 = \sum_{k=1}^{K} \sum_{h} \|W^{(h)}\|_0, \qquad f_2 = \sum_{k=1}^{K} \frac{|D_k|}{|D|} \mathbb{E}_{(x,y)\sim D_k}[F(x, y; W)], \tag{17}$$

Here, W denotes the model parameters of the neural network, and h represents the number of hidden layers. K indicates the total number of clusters obtained after client clustering. For each cluster k, D_k is the associated dataset, with $|D_k|$ and $|D|$ representing the cluster-specific and total number of samples, respectively. In the clustered federated learning scenario, f_2 denotes the global loss function. Specifically, it is calculated as the weighted sum of local losses from all clusters, with weights determined by the data scale. This formulation ensures that f_2 accurately reflects the performance of the clustered model across the entire data distribution. It can thus be seen that, although the problem in this subsection is similar to a multitask problem, the actual objective function is globally consistent.

https://doi.org/10.3390/app16105094

3.2.2. Inter-Population Knowledge Transfer

To address the optimization difficulty caused by data heterogeneity in federated learning, this paper proposes a clustered federated multi-population particle swarm optimization algorithm (CF-MPPSO). In this framework, each client cluster corresponds to one particle population, which independently performs parameter search and generates local nondominated candidate solutions. Subsequently, a global evaluation is conducted through cross-cluster combinations, allowing each population to be updated according to the resulting global Pareto solutions. This method enables the characteristics of each cluster of data to be preserved while realizing global collaborative optimization. Suppose that there are K client clusters in the federated system: $C_1, C_2, \ldots, C_K$, and each cluster corresponds to one particle population: $P_1, P_2, \ldots, P_K$. The ith particle in cluster k is denoted as $X_{k,i} = (x_{k,i1}, x_{k,i2}, \ldots, x_{k,id})$. Here, d represents the dimension of the search space. The evaluation of this particle on cluster C_k is performed as follows:

$$\mathbf{F}_k(X_{k,i}) = (f_{k,1}(X_{k,i}), f_{k,2}(X_{k,i})). \tag{18}$$

Let each cluster C_k maintain a population P_k. Nondominated solutions are first identified independently within each cluster:

$$A_k = \text{nondominated}(P_k). \tag{19}$$

and the cluster-level candidate solution set is obtained:

$$A_k = \{X^*_{k,1}, X^*_{k,2}, \ldots\}, \tag{20}$$

According to Equation (16), the weight of each cluster is determined by the proportion of the number of client samples:

$$w_k = \frac{|D_k|}{|D|}, \qquad \sum_{k=1}^{K} w_k = 1. \tag{21}$$

To construct a global joint solution, one nondominated solution is selected from each cluster. If $X^*_{1,i_1}, X^*_{2,i_2}, \ldots, X^*_{K,i_K}$ are chosen, then the joint solution is as follows:

$$S = X^*_{1,i_1} \oplus X^*_{2,i_2} \oplus \cdots \oplus X^*_{K,i_K}, \tag{22}$$

where $\oplus$ denotes the parameter vector concatenation operation, and $i_k \in \{1, 2, \ldots, |A_k|\}$. By traversing all nondominated combinations, the joint solution set is as follows:

$$\mathcal{S} = \left\{ X^*_{1,i_1} \oplus X^*_{2,i_2} \oplus \cdots \oplus X^*_{K,i_K} \;\middle|\; X^*_{k,i_k} \in A_k,\ k = 1, \ldots, K \right\}, \tag{23}$$

whose size is

$$|\mathcal{S}| = \prod_{k=1}^{K} |A_k|. \tag{24}$$

After global objective evaluation and nondominated sorting on the joint particle set, the global nondominated archive is obtained as

$$A^{\text{global}} = \text{nondominated}(\mathcal{S}), \tag{25}$$

The components of solutions in A^{global} are then projected back to their corresponding clusters to update the local personal best and global best particles. It follows that every

https://doi.org/10.3390/app16105094

global Pareto solution must be formed by combining nondominated solutions drawn from the individual clusters. This is the central knowledge-transfer mechanism of CF-MPPSO.

3.2.3. Clustered Multi-Objective Particle Swarm Federated Learning

The pseudocode in Algorithm 3 enters the main iterative process from line 3 onward. Lines 4–10 mainly complete particle evaluation within each cluster. In this stage, the algorithm processes particles cluster by cluster and obtains objective values according to the corresponding cluster-specific data environment. A global evaluation is formed through cross-cluster candidate-solution combinations. Subsequently, each population is updated according to the resulting global Pareto solutions. In line 20, $\pi_k(\cdot)$ denotes the projection from the global joint archive onto the decision variables associated with the kth cluster.

Algorithm 3 CF-MPPSO for Clustered Federated Learning

Input: client clusters C_k and corresponding particle populations $P_1, P_2, \ldots, P_K$
Output: global nondominated solution set A^{global}

1: Initialize particle positions and velocities for all populations
2: Initialize personal bests *Pbest* and global bests *Gbest*
3: **for** $t = 1$ to T **do**
4: **for** $k = 1$ to K **do**
5: **for** each particle $X_{k,i}$ in cluster C_k **do**
6: Execute clustered federated training on C_k
7: Compute objective vector $\mathbf{F}_k(X_{k,i}) = (f_{k,1}, f_{k,2}, \ldots, f_{k,m})$
8: **end for**
9: Perform Pareto nondominated sorting; obtain $A_k = \text{nondominated}(P_k)$
10: **end for**
11: Initialize joint solution set $S \leftarrow \emptyset$
12: **for** each combination $X^*_{1,i_1}, X^*_{2,i_2}, \ldots, X^*_{K,i_K}$ **do**
13: Construct joint solution $S = X^*_{1,i_1} \oplus X^*_{2,i_2} \oplus \cdots \oplus X^*_{K,i_K}$
14: Evaluate joint objective value $F_j(S) = \sum_{k=1}^{K} w_k f_{k,j}(X^*_{k,i_k})$
15: Add joint solution to candidate set
16: **end for**
17: Compute $A^{global} = \text{nondominated}(S)$
18: **for** $k = 1$ to K **do**
19: **for** each particle $X_{k,i}$ **do**
20: Project global archive: $Gbest_k = \pi_k(A^{global})$
21: Update velocity: $V^{t+1}_{k,i} = \omega V^t_{k,i} + c_1 r_1 (Pbest_{k,i} - X_{k,i}) + c_2 r_2 (Gbest_k - X_{k,i})$
22: Update position: $X^{t+1}_{k,i} = X^t_{k,i} + V^{t+1}_{k,i}$
23: **end for**
24: **end for**
25: **end for**
26: **Return** A^{global}

3.3. Computational Complexity Analysis

In each iteration, the algorithm first evaluates particles through clustered federated training. If the cost of one federated evaluation is denoted by $\mathcal{C}_{\text{FL}}$, then the complexity of cluster-level evaluation is $O(KN\mathcal{C}_{\text{FL}})$. Next, Pareto nondominated sorting is performed among the N particles in each cluster, and the resulting complexity is $O(KmN^2)$. Let the number of nondominated solutions in cluster k be M_k. Then the number of joint solutions is $|\mathcal{S}| = \prod_{k=1}^{K} M_k$. Therefore, constructing the joint solution set costs $O(\prod_{k=1}^{K} M_k)$,

https://doi.org/10.3390/app16105094

while global objective evaluation and nondominated sorting cost $O\left(m\prod_{k=1}^{K}M_k\right)$ and $O\left(m\left(\prod_{k=1}^{K}M_k\right)^2\right)$, respectively. Hence, the total complexity of one iteration is as follows:

$$O\left(KNC_{\mathrm{FL}}+KmN^2+|\mathcal{S}|+m|\mathcal{S}|^2\right), \tag{26}$$

If the maximum number of iterations is T, then the overall complexity becomes

$$O\left(T\left[KNC_{\mathrm{FL}}+KmN^2+|\mathcal{S}|+m|\mathcal{S}|^2\right]\right). \tag{27}$$

Unlike the clustered method, the non-clustered PSO federated learning framework maintains only a single candidate population in the global search space. Their complexity can be approximated as follows:

$$O(TKNC_{\mathrm{FL}}+mK^2N^2). \tag{28}$$

In contrast, the method in this section exhibits higher search complexity. This is primarily due to the introduction of intra-cluster nondominated sorting, cross-cluster joint-solution construction, and global Pareto filtering. While the clustered federated algorithm involves additional optimization steps, its evaluation process remains efficient. Although the clustered method introduces additional optimization overhead, it does not require extra model-training stages beyond those already involved in federated architecture search. In federated architecture search, the computational bottleneck is almost always model training, and it can be seen that CF-MPPSO does not additionally introduce a model-training process. Through this additional computational overhead, CF-MPPSO explicitly models the data heterogeneity among users in different clusters. Furthermore, it improves the architecture search capability in non-IID scenarios through a joint optimization mechanism.

4. Experiments and Analysis

This section first introduces the datasets and experimental settings and then reports the comparative results under IID and non-IID scenarios.

4.1. Datasets

The proposed method is evaluated on the Modified National Institute of Standards and Technology (MNIST) [40] and the Canadian Institute for Advanced Research 10-class (CIFAR10) [41] datasets. MNIST comprises handwritten digits across 10 classes (0–9), consisting of 60,000 training and 10,000 test samples. Each sample is represented as a 28×28 single-channel grayscale image. In contrast, CIFAR10 is a more complex dataset of natural color images, containing 50,000 training and 10,000 test 32×32 red–green–blue (RGB) images. These images are evenly distributed across 10 categories, with each category containing 6000 samples. To address the non-IID data-distribution scenario, we partition data with different labels into groups and randomly assign them to users. In this experiment, clients are partitioned into three categories, and each category is assigned non-overlapping data classes to simulate the non-IID setting. In this experiment, the particle swarm optimization federated learning (PSOFL) algorithm (Algorithm 2) and CF-MPPSO (Algorithm 3) are employed to address the IID and non-IID data-distribution scenarios, respectively.

4.2. Parameter Settings

All experimental settings adopted in this framework will be described in this subsection. The model architecture is identical to that of a general deep neural network, which is composed of convolutional layers (Conv), pooling layers, and fully connected layers. Each

https://doi.org/10.3390/app16105094

convolutional layer consists of a Conv–ReLU–BN structure, where ReLU represents the rectified linear unit activation function and BN represents batch norm.

In the federated learning configuration, $M = 10$ clients and 100 communication rounds are utilized. The PSO algorithm is configured with a population of 32 particles evolving over 40 generations. Key hyperparameters include acceleration constants $c_1 = c_2 = 2$ and an inertia weight w in the range of $[0.4, 0.9]$, following widely adopted practices in the PSO literature.

The multi-objective evolutionary federated learning (MEFL) algorithm using sparse evolution [42] serves as the primary comparative baseline. Detailed parameter search ranges are presented in Table 1. For all experiments, the evolutionary algorithm optimizes the learning rate within a range of 0.001 to 0.3. The network architecture comprises 1 to 3 randomly selected convolutional layers, each featuring a kernel size of 3 or 5. Correspondingly, the upper limits for the number of channels across these layers are set to 64, 128, and 256, respectively. If only 2 convolutional layers are selected, the third layer is omitted. This ensures that the corresponding portion of the encoding remains unutilized. Following the convolutional part, the number of fully connected layers is set to 1 or 2, with neuron ranges of 512–1024 and 256–512, respectively. Similar to the convolutional architecture, the second layer configuration is ignored if the layer count is 1. To ensure reliability, all experiments were performed three times, and the resulting averages were calculated. All experiments were implemented in Python 3.8 with PyTorch and conducted on a server running Ubuntu 20.04, equipped with a 16-core 2.6 GHz CPU and 128 GB RAM. The GPU used for deep learning experiments was an NVIDIA GeForce RTX 4090 Ti GPU (NVIDIA Corporation, Santa Clara, CA, USA).

Table 1. Encoded parameters in the search space.

Parameter	PSOFL	MEFL
Population size	32	32
Iterations	40	40
Kernel size	3 or 5	3 or 5
Conv layer 1 channels	1–64	1–64
Conv layer 2 channels	1–128	1–128
Number of fully connected layers	1 or 2	1 or 2
Fully connected layer 1 neurons	512–1024	512–1024
Fully connected layer 2 neurons	256–512	256–512
Learning rate	0.001–0.3	0.001–0.3
Momentum	0.1–0.9	/

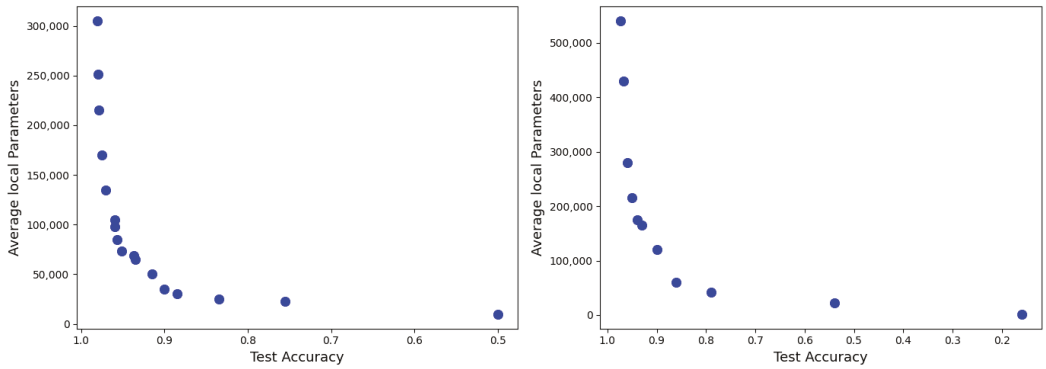
In our work, the number of clusters K for the clustered federated learning experiments is determined by the construction of the synthetic datasets and thus treated as given in the optimization studies. When the true cluster structure of clients is known or defined by data generation design, using that value of K allows us to focus on evaluating the optimization behavior of CF-MPPSO itself under heterogeneous federated settings. We do not explore a wide range of fixed K values because doing so would mix the cluster selection problem with the optimizer evaluation problem, potentially confounding the interpretation of optimization performance.

4.3. Experimental Results and Analysis

The IID results on the MNIST and CIFAR10 datasets are shown in Figures 4 and 5. Each point represents a nondominated solution of the model. In performance measurement, we are more concerned with points that have high accuracy or the best trade-offs. Therefore, we separately focus on the high-accuracy points *High*1 and *High*2, as well as the two knee

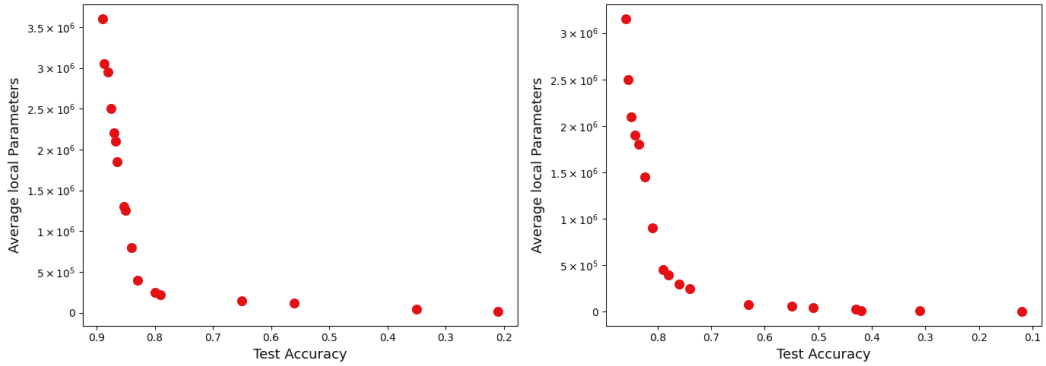
https://doi.org/10.3390/app16105094

points *Knee*1 and *Knee*2. The best results are highlighted in bold, where higher Acc and lower Params indicate better performance.



(**a**) PSOFL (**b**) MEFL

Figure 4. Pareto fronts on the MNIST dataset under IID data distribution.



(**a**) PSOFL (**b**) MEFL

Figure 5. Pareto fronts on the CIFAR10 dataset under IID data distribution.

MNIST is a handwritten digit recognition dataset with a relatively small scale and simple structure. The classification task is relatively simple due to its lower feature dimension. Because of its low training cost, MNIST is often used as a benchmark dataset to verify the effectiveness of optimization algorithms. From the results shown in Table 2, we can draw the following main conclusions:

- For the IID data distribution on the MNIST dataset, the model optimized by the proposed algorithm achieves a peak accuracy of 98.1% with 305,946 parameters. Similarly, the MEFL algorithm yields an accuracy of 96.8% while maintaining a larger parameter scale of 540,144. These results demonstrate a slight improvement in overall accuracy alongside a significant 43.4% reduction in parameter scale.
- At the two knee points on the MNIST dataset, the proposed method reaches 93.5% accuracy with a parameter count of 73,276. In contrast, the MEFL comparison method achieves an accuracy of 90.1% with 123,978 parameters. These trade-off results indicate that the proposed method reduces the model scale by 50.5% while simultaneously achieving an improvement in accuracy.

The CIFAR10 dataset possesses a more complex image structure and more diverse inter-category characteristics. Consequently, differences between optimization strategies regarding convergence speed, stability, and final accuracy become more apparent. Based on the results, the following main conclusions can be drawn:

- On the CIFAR10 dataset under the IID setting, the highest accuracy among the representative solutions obtained by the proposed method is 89.4%, while the correspond-

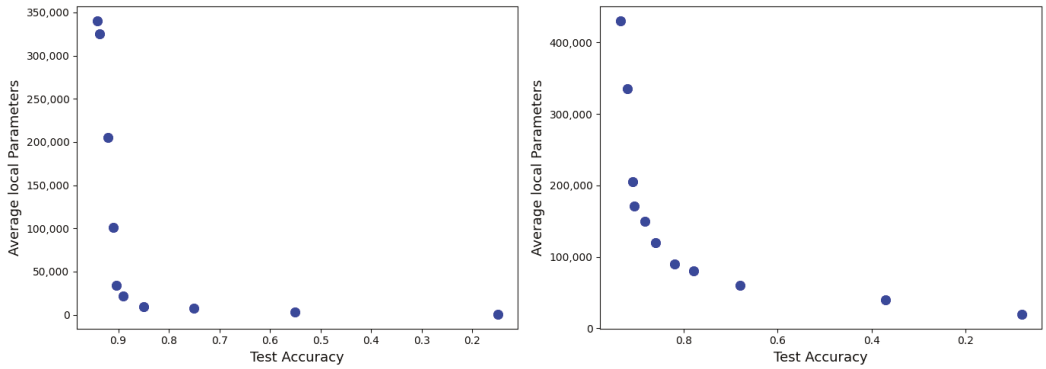
ing value for MEFL is 86.5%, giving an absolute improvement of 2.9 percentage points. However, no reduction in parameter count is observed at the two *High* points.

- At the two knee points for the CIFAR10 dataset, the proposed method reaches accuracies of 82.9% and 84.0%, corresponding to 407,264 and 853,036 parameters, respectively. Compared with the two *knee* points of MEFL, the proposed algorithm demonstrates improved overall accuracy. Specifically, one knee solution obtained by the proposed method Pareto dominates its counterpart from MEFL.

Table 2. Comparison with the mainstream method under IID data distribution.

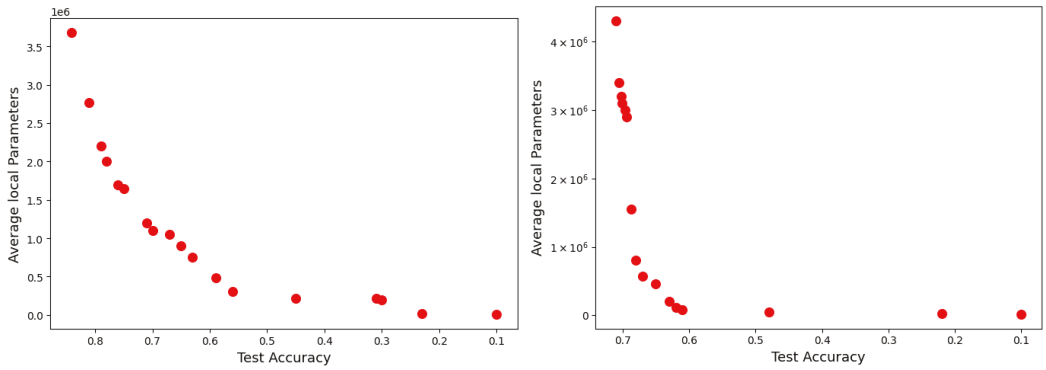
Dataset	MNIST		CIFAR10	
	Acc	**Params**	**Acc**	**Params**
High1 (PSOFL)	**98.1 ± 0.45**	305,946	**89.4 ± 0.67**	3,685,016
High2 (PSOFL)	97.6 ± 0.34	**250,144**	88.6 ± 0.71	3,054,348
High1 (MEFL)	97.4 ± 0.63	546,940	86.5 ± 0.56	3,175,180
High2 (MEFL)	96.8 ± 0.36	430,144	85.3 ± 0.41	**2,534,718**
Knee1 (PSOFL)	**93.5 ± 0.79**	65,276	82.9 ± 0.70	**407,264**
Knee2 (PSOFL)	91.5 ± 0.31	**56,242**	**84.0 ± 0.41**	853,036
Knee1 (MEFL)	90.1 ± 0.37	123,978	80.8 ± 0.37	576,734
Knee2 (MEFL)	86.2 ± 0.41	61,676	79.4 ± 0.34	461,782

Similarly, the non-IID results on the MNIST and CIFAR10 datasets are shown in Figures 6 and 7. The local parameter values have become the weighted average of the model parameters of each cluster.



(**a**) CF-MPPSO (**b**) MEFL

Figure 6. Pareto fronts on the MNIST dataset under non-IID data distribution.



(**a**) CF-MPPSO (**b**) MEFL

Figure 7. Pareto fronts on the CIFAR10 dataset under non-IID data distribution.

From the results shown in Table 3, we can draw the following main conclusions:

- On the MNIST dataset, the highest accuracy of the model optimized by the proposed algorithm for the non-IID data-distribution scenario reaches 94.2%, which is a slight improvement compared with the accuracy index of the MEFL algorithm. However, at the two knee points, the proposed algorithm greatly reduces the required number of parameters while maintaining accuracy.
- On the CIFAR10 dataset, the highest accuracy of the model optimized by the proposed algorithm for the non-IID data-distribution scenario reaches 84.1%, which is a substantial improvement of 17.6% compared with the accuracy index of the MEFL algorithm. At the two knee points, significant improvements in accuracy are obtained despite having similar parameter counts. Furthermore, certain solutions in these regions are shown to dominate the mainstream MEFL algorithm.

Table 3. Comparison with the mainstream method under non-IID data distribution.

Dataset	MNIST		CIFAR10	
	Acc	**Params**	**Acc**	**Params**
High1 (CF-MPPSO)	**94.2 ± 0.45**	345,186	**84.1 ± 0.35**	3,685,016
High2 (CF-MPPSO)	93.7 ± 0.42	**325,788**	81.1 ± 0.40	**2,768,934**
High1 (MEFL)	93.6 ± 0.68	435,180	71.5 ± 0.54	4,335,682
High2 (MEFL)	92.1 ± 0.65	335,178	70.2 ± 0.55	3,424,658
Knee1 (CF-MPPSO)	90.4 ± 0.35	37,408	**78.9 ± 0.52**	577,264
Knee2 (CF-MPPSO)	89.3 ± 0.46	**27,356**	77.4 ± 0.70	**354,672**
Knee1 (MEFL)	**92.7 ± 0.42**	144,908	69.6 ± 0.35	576,534
Knee2 (MEFL)	89.9 ± 0.48	130,764	68.7 ± 0.32	461,826

We observed a noticeable difference in the relative performance advantage of CF-MPPSO over MEFL between the two datasets: on CIFAR-10 non-IID settings, CF-MPPSO shows a large improvement (17.6%), whereas, on MNIST non-IID at high points, the gain is less than 1%. This discrepancy can be attributed to the inherent complexity and feature diversity of the datasets. The MNIST dataset consists of grayscale images of handwritten digits with relatively simple and well-separable classes, which many models can handle effectively, leading to high baseline performance and less room for further optimization. In contrast, CIFAR-10 comprises color images of real-world object categories with substantial intra-class variability and more complex visual patterns, making the classification task inherently harder and more heterogeneous across client distributions. Under such complex non-IID conditions, the cluster-specific search and collaborative exploration mechanisms in CF-MPPSO provide a stronger benefit by better adapting to localized data characteristics and enhancing global search diversity, resulting in a more pronounced performance advantage.

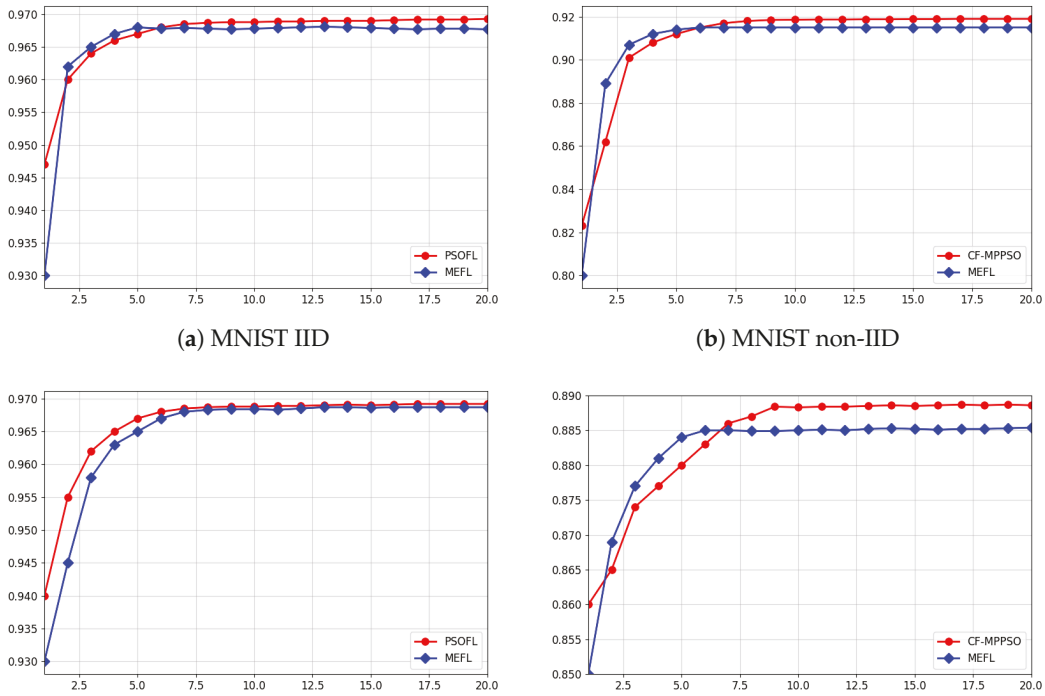
The experimental results demonstrate that the genetic knowledge-transfer mechanism of the PSO algorithm is highly effective in multi-population optimization. This contribution is particularly significant when addressing the data-heterogeneity problem in federated learning. Meanwhile, maintaining multiple models on the client side significantly reduces the computational burden during the optimization process. Because of computational resource limitations, the method in this paper is restricted to 10 clients. CF-MPPSO thus still operates under conditions of incomplete data information. Nonetheless, the inherent advantages of the framework will be further revealed as user and group counts increase.

https://doi.org/10.3390/app16105094

Hypervolume Indicator

Here, we utilize the hypervolume indicator to discuss the convergence characteristics of the compared algorithms. HV is a performance metric used to measure the region covered by an approximate Pareto front. It is widely adopted in multi-objective optimization (MOO) problems due to its robust evaluation. Specifically, a larger HV value reflects better performance of the approximate front in terms of both diversity and coverage.

The evolution of the HV indicator is shown in Figure 8. One advantage of the PSO algorithm is that, when the number of particles is relatively large, it can start quickly. Although the proposed method requires more iterations before convergence than MEFL, the overall result is significantly improved compared with MEFL.



(**a**) MNIST IID (**b**) MNIST non-IID

(**c**) CIFAR10 IID (**d**) CIFAR10 non-IID

Figure 8. Comparison of HV indicators: (**a**) MNIST IID data distribution; (**b**) MNIST non-IID data distribution; (**c**) CIFAR10 IID data distribution; (**d**) CIFAR10 non-IID data distribution.

4.4. Ablation Study

To better understand the contribution of each component in the proposed framework, we conduct an ablation study by progressively removing key modules from the full method. Specifically, we consider four variants: (1) base FL, which uses standard federated learning without clustered optimization or PSO-based search; (2) PSOFL, which introduces PSO-based optimization without client clustering or inter-population knowledge transfer; (3) clustered PSO w/o knowledge transfer, which incorporates clustered federated learning and PSO-based optimization but disables the inter-population knowledge transfer mechanism; and (4) the full CF-MPPSO method.

The results are reported in Table 4. It can be observed that introducing PSO-based optimization already improves the performance over the basic federated baseline, indicating that automatic search of architecture and hyperparameters is beneficial. When clustered federated learning is further introduced, the performance is improved again, suggesting that grouping clients with similar data characteristics helps to alleviate the negative impact of heterogeneity. Finally, the full CF-MPPSO achieves the best overall performance, which shows that the inter-population knowledge transfer mechanism further enhances search efficiency and robustness under non-IID settings.

https://doi.org/10.3390/app16105094

Table 4. Ablation study of representative Pareto-optimal solutions obtained by different variants of the proposed framework on CIFAR10 under the non-IID setting.

Variant	Accuracy (%)	Params (M)	HV
FL	71.5 ± 0.4	4.30	N/A
PSOFL	73.1 ± 0.3	3.50	0.650
Clustered PSO w/o Transfer	77.9 ± 0.4	2.69	0.760
CF-MPPSO	81.1 ± 0.4	2.77	0.889

These results demonstrate that all three components contribute positively to the final performance, while the combination of clustered optimization and inter-population knowledge transfer provides the largest gain.

As shown in Table 4, the reported results correspond to representative solutions selected from the Pareto-optimal sets obtained by different variants rather than arbitrary single points. Therefore, the comparison is intended to illustrate the practical quality of the trade-off solutions generated by each method.

From the table, the PSOFL variant improves the representative solution over the basic FL baseline, increasing the accuracy from 71.5% to 73.1% while reducing the model size from 4.3 M to 3.5 M. From the previous section, we know that, compared with the MFEA algorithm, PSO provides a more effective search process for discovering competitive trade-off solutions under the current setting. However, the improvement in accuracy brought by changing the search strategy alone is relatively limited.

A more substantial gain is observed after introducing clustered federated learning. Compared with PSOFL, the clustered PSO w/o transfer variant further improves the representative accuracy from 73.1% to 73.6% while significantly reducing the parameter size to 2.69 M. This indicates that clustering is the more critical factor for improving predictive performance since it better accommodates heterogeneous and non-IID client distributions.

The full CF-MPPSO achieves the best representative solution, reaching 81.1% accuracy with 2.77 M parameters and the highest HV value of 0.889. This result shows that, on top of clustered optimization, the inter-population knowledge transfer mechanism further strengthens the search and enables the algorithm to obtain a richer set of high-quality Pareto-optimal solutions.

Overall, the ablation study suggests that PSO mainly contributes to optimization efficiency and solution discovery, whereas clustering is the key factor behind the major accuracy improvement. The knowledge transfer mechanism further enhances the quality of the obtained representative Pareto-optimal solutions.

5. Conclusions and Future Work

In this paper, we investigated the application of particle swarm optimization to federated learning and proposed a multi-objective optimization framework for both IID and heterogeneous data environments. Traditional particle swarm federated learning methods usually rely on a single-population search strategy, which is prone to premature convergence and local optima in complex search spaces, especially under non-IID data distributions. As a result, their ability to handle multi-objective optimization problems in federated learning remains limited.

To address this issue, we developed a clustered federated learning method based on multi-population particle swarm optimization, namely CF-MPPSO. By mapping different client clusters to multiple subpopulations, the proposed framework enables population-specific evolution under local data distributions while introducing collaborative optimization across populations. This mechanism improves search diversity, global exploration capability, and robustness in heterogeneous federated learning scenarios.

https://doi.org/10.3390/app16105094

The experimental results on two representative datasets under both IID and non-IID settings demonstrate that the proposed methods outperform existing federated learning multi-objective optimization approaches. In particular, PSOFL achieves better performance in IID scenarios, while CF-MPPSO shows stronger effectiveness under non-IID conditions, providing superior trade-offs between predictive accuracy and model complexity. These results verify the effectiveness of the proposed framework for complex federated learning optimization tasks.

Although the proposed framework has shown promising performance, several issues still deserve further investigation. In particular, more adaptive clustering strategies and more realistic heterogeneous federated settings should be incorporated into the optimization framework. Moreover, as federated learning techniques continue to evolve, scalability to large-scale applications involving massive numbers of clients and high-dimensional data is a critical direction for future work. Recent studies have highlighted the importance of designing federated systems that can operate efficiently at scale, addressing communication, architectural, and resource heterogeneity challenges to enable deployment in real-world Big Data and edge environments.

Author Contributions: Conceptualization, C.J. and M.G.; Methodology, Z.L.; Software, Z.L.; Formal analysis, C.J.; Investigation, C.J. and M.G.; Resources, C.J.; Data curation, C.J.; Writing—original draft, Z.L.; Writing—review & editing, Z.L.; Visualization, C.J.; Supervision, H.L. and M.G.; Project administration, H.L.; Funding acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China: 62576259.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Shokri, R.; Shmatikov, V. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*; Association for Computing Machinery: New York, NY, USA, 2015; pp. 1310–1321.
2. Dean, J.; Corrado, G.; Monga, R.; Chen, K.; Devin, M.; Mao, M.; Ranzato, M.; Senior, A.; Tucker, P.; Yang, K.; et al. Large scale distributed deep networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1–9.
3. Wang, F.; Wang, Y.; Chen, Y.; Tian, Y.; Qi, H.; Wang, X.; Zhang, W.; Zhang, J.; Yuan, Y. Federal ecology: From federated data to federated intelligence. *Chin. J. Intell. Sci. Technol.* **2020**, *2*, 305–311.
4. Yang, Q.; Tong, Y.; Wang, Y.; Fan, L.; Wang, W.; Chen, L.; Wang, W.; Kang, Y. A survey of federated learning algorithms in collective intelligence. *Chin. J. Intell. Sci. Technol.* **2022**, *4*, 29–44.
5. Li, N.; Ma, L.; Yu, G.; Xue, B.; Zhang, M.; Jin, Y. Survey on evolutionary deep learning: Principles, algorithms, applications, and open issues. *ACM Comput. Surv.* **2023**, *56*, 41. [CrossRef]
6. Liu, Y.; Sun, Y.; Xue, B.; Zhang, M.; Yen, G.G.; Tan, K.C. A survey on evolutionary neural architecture search. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *34*, 550–570. [CrossRef]
7. Mao, Y.; You, C.; Zhang, J.; Huang, K.; Letaief, K.B. A survey on mobile edge computing: The communication perspective. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2322–2358. [CrossRef]
8. Nayak, J.; Swapnarekha, H.; Naik, B.; Dhiman, G.; Vimal, S. 25 years of particle swarm optimization: Flourishing voyage of two decades. *Arch. Comput. Methods Eng.* **2023**, *30*, 1663–1725. [CrossRef]
9. Shami, T.M.; El-Saleh, A.A.; Alswaitti, M.; Al-Tashi, Q.; Summakieh, M.A.; Mirjalili, S. Particle swarm optimization: A comprehensive survey. *IEEE Access* **2022**, *10*, 10031–10061. [CrossRef]

10. Fang, J.; Liu, W.; Chen, L.; Lauria, S.; Miron, A.; Liu, X. A Survey of Algorithms, Applications and Trends for Particle Swarm Optimization. *Int. J. Netw. Dyn. Intell.* **2023**, *2*, 24–50. [CrossRef]
11. Xiong, P.; Zhu, T.; Wang, X. A Survey on Differential Privacy and Applications. *Chin. J. Comput.* **2014**, *37*, 101–122.
12. Bogetoft, P.; Christensen, D.L.; Damgård, I.; Geisler, M.; Jakobsen, T.; Krøigaard, M.; Nielsen, J.D.; Nielsen, J.B.; Nielsen, K.; Pagter, J.; et al. Secure multiparty computation goes live. In *Financial Cryptography and Data Security* ; Springer: Berlin/Heidelberg, Germany, 2009; pp. 325–343.
13. Nasr, M.; Shokri, R.; Houmansadr, A. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP)*; IEEE: New York, NY, USA, 2019; pp. 739–753.
14. Su, H.; Zhao, Y.; Zhang, W.; Zhang, H.; Huang, S.; Lu, M.; Zhou, X. Secure and Verifiable Edge-Federated Learning with Homomorphic Encryption and a Trusted Execution Environment for UAV Communication. *Electronics* **2026**, *15*, 1029. [CrossRef]
15. Liu, J.; Huang, J.; Zhou, Y.; Li, X.; Ji, S.; Xiong, H.; Dou, D. From distributed machine learning to federated learning: A survey. *Knowl. Inf. Syst.* **2022**, *64*, 885–917. [CrossRef]
16. Koutsoubis, N.; Waqas, A.; Yilmaz, Y.; Ramachandran, R.P.; Schabath, M.B.; Rasool, G. Privacy-preserving federated learning and uncertainty quantification in medical imaging. *Radiol. Artif. Intell.* **2025**, *7*, e240637. [CrossRef] [PubMed]
17. Verbraeken, J.; Wolting, M.; Katzy, J.; Kloppenburg, J.; Verbelen, T.; Rellermeyer, J.S. A survey on distributed machine learning. *ACM Comput. Surv.* **2020**, *53*, 30. [CrossRef]
18. Hao, W.; El-Khamy, M.; Lee, J.; Zhang, J.; Liang, K.J.; Chen, C.; Duke, L.C. Towards fair federated learning with zero-shot data augmentation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*; IEEE: New York, NY, USA, 2021; pp. 3310–3319.
19. Pfitzner, B.; Steckhan, N.; Arnrich, B. Federated learning in a medical context: A systematic literature review. *ACM Trans. Internet Technol.* **2021**, *21*, 50. [CrossRef]
20. Kim, J.; Park, T.; Kim, H.; Kim, S. Federated learning for face recognition. In *2021 IEEE International Conference on Consumer Electronics (ICCE)*; IEEE: New York, NY, USA, 2021; pp. 1–2.
21. Li, L.; Fan, Y.; Tse, M.; Lin, K.Y. A review of applications in federated learning. *Comput. Ind. Eng.* **2020**, *149*, 106854. [CrossRef]
22. Dembani, R.; Karvelas, I.; Akbar, N.A.; Rizou, S.; Tegolo, D.; Fountas, S. Agricultural data privacy and federated learning: A review of challenges and opportunities. *Comput. Electron. Agric.* **2025**, *232*, 110048. [CrossRef]
23. Fallah, A.; Mokhtari, A.; Ozdaglar, A. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 3557–3568.
24. Shi, L.; Li, Z.; Ding, X.; Zhu, Y.; Yao, X. A Federated CLIP Fine-Tuning Method Based on Optimal Transport and Dual Prompt Personalization. *Electronics* **2026**, *15*, 972. [CrossRef]
25. Xu, J.; Du, W.; Jin, Y.; He, W.; Cheng, R. Ternary compression for communication-efficient federated learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *33*, 1162–1176. [CrossRef]
26. Gao, Y.; Gong, M.; Xie, Y.; Qin, A.K.; Pan, K.; Ong, Y.S. Multiparty dual learning. *IEEE Trans. Cybern.* **2022**, *53*, 2955–2968. [CrossRef]
27. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
28. Chen, J.; Pan, X.; Monga, R.; Bengio, S.; Jozefowicz, R. Revisiting distributed synchronous SGD. *arXiv* **2016**, arXiv:1604.00981.
29. Goodfellow, I.J.; Vinyals, O.; Saxe, A.M. Qualitatively characterizing neural network optimization problems. *arXiv* **2014**, arXiv:1412.6544.
30. Li, Q.; Kim, B.M. Clustering approach for hybrid recommender system. In *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)*; IEEE: New York, NY, USA, 2003; pp. 33–38.
31. Sarwar, B.M.; Karypis, G.; Konstan, J.; Riedl, J. Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the 5th International Conference on Computer and Information Technology*; IEEE Computer Society: Washington, DC, USA, 2002; Volume 1, pp. 291–324.
32. Smith, V.; Chiang, C.K.; Sanjabi, M.; Talwalkar, A.S. Federated multi-task learning. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 4427–4437.
33. Sattler, F.; Müller, K.R.; Samek, W. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 3710–3722. [CrossRef] [PubMed]
34. Jiang, Y.; Konečnỳ, J.; Rush, K.; Kannan, S. Improving federated learning personalization via model agnostic meta learning. *arXiv* **2019**, arXiv:1909.12488.
35. Li, Z.; Shao, J.; Mao, Y.; Wang, J.H.; Zhang, J. Federated learning with gan-based data synthesis for non-iid clients. In *Trustworthy Federated Learning: First International Workshop*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 17–32.

36. Sattler, F.; Wiedemann, S.; Müller, K.R.; Samek, W. Robust and communication-efficient federated learning from non-iid data. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *31*, 3400–3413. [CrossRef]
37. Li, T.; Sahu, A.K.; Talwalkar, A.; Smith, V. Federated learning: Challenges, methods, and future directions. *IEEE Signal Process. Mag.* **2020**, *37*, 50–60. [CrossRef]
38. Ghosh, A.; Chung, J.; Yin, D.; Ramchandran, K. An efficient framework for clustered federated learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 19586–19597. [CrossRef]
39. Kennedy, J.; Eberhart, R. Particle swarm optimization. In *Proceedings of ICNN'95—International Conference on Neural Networks*; IEEE: New York, NY, USA, 1995; Volume 4, pp. 1942–1948.
40. Deng, L. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Process. Mag.* **2012**, *29*, 141–142. [CrossRef]
41. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; University of Toronto: Toronto, ON, Canada, 2009.
42. Zhu, H.; Jin, Y. Multi-Objective Evolutionary Federated Learning. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 1310–1322. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

SupGAN: A General Super-Resolution GAN-Promoting Training Method

Tao Wu [1], Shuo Xiong [2,*], Qiuhang Chen [1], Huaizheng Liu [1], Weijun Cao [1] and Haoran Tuo [1]

[1] School of Software Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; wutao@hust.edu.cn (T.W.); nickcqh@hust.edu.cn (Q.C.); liuhuaizheng@hust.edu.cn (H.L.); caoweijun@hust.edu.cn (W.C.); thr@hust.edu.cn (H.T.)

[2] School of Journalism and Information Communication, Huazhong University of Science and Technology, Wuhan 430074, China

* Correspondence: xiongshuo@hust.edu.cn

Abstract

An image super-resolution (SR) method based on Generative Adversarial Networks (GANs) has achieved impressive results in terms of visual performance. However, the weights of loss functions in these methods are usually set to fixed values manually, which cannot fully adapt to different datasets and tasks, and may result in a decrease in the perceptual effect of the SR images. To address this issue and further improve visual quality, we propose a perception-driven SupGAN, which improves the generator and loss function of GAN-based image super-resolution models. The generator adopts multi-scale feature extraction and fusion to restore SR images with diverse and fine textures. We design a network-training method based on the proportion of high-frequency information in images (BHFTM), which utilizes the proportion of high-frequency information in images obtained through the Canny operator to set the weights of the loss function. In addition, we employ the four-patch method to better simulate the degradation of complex real-world scenarios. We extensively test our method and compare it with recent SR methods (BSRGAN, Real-ESRGAN, RealSR, SwinIR, LDL, etc.) on different types of datasets (OST300, 2020track1, RealWorld38, BSDS100 etc.) with a scaling factor of $\times 4$. The results show that the NIQE metric improves, and also demonstrate that SupGAN can generate more natural and fine textures while suppressing unpleasant artifacts.

Keywords: perception design; image super resolution; Generative Adversarial Network; degradation model; loss function

1. Introduction

The purpose of single-image super-resolution (SISR) is to convert a low-resolution (LR) into a high-resolution (HR) image, and it aims to enhance the resolution and quality of the image to make it clearer and with more details. Traditionally, it is challenging to convert an LR image into a HR image, and the main methods are interpolation-based [1,2] and reconstruction-based [3–5]. Currently, SISR methods typically employ deep learning-based models, such as Convolutional Neural Networks (CNNs). These learning-based methods are trained to extract features from the low-resolution image and map them to the HR image, which are then used to reconstruct the final output.

Since Generative Adversarial Networks (GANs) [6] were introduced in the field of super-resolution, GANs have been widely used in this field and have shown great potential. SRGAN [7] employs SRResNet as the generator within its GAN framework and introduces a perceptual loss that blends content loss with adversarial loss to train the network. ESRGAN [8] improves the adversarial loss using a relativistic GAN [9] on the basis of SRGAN and enhances the perceptual loss by using features before activation. Real-ESRGAN [10,11] also uses perceptual loss, adversarial loss, and content loss as loss functions.

Most super-resolution models use composite loss functions composed of multiple loss functions, whose weights are usually manually set and fixed during training. However, there are differences in the proportion of high-frequency information among real training samples. For GAN-based super-resolution methods, the weights of the loss functions used are constant, resulting in the equal treatment of all training samples. For images with a higher proportion of low-frequency information, their low-frequency information can sometimes cause the training to proceed in the wrong direction in terms of perception. This can lead to problems such as insufficient recovery of detailed information by the generator, limited discrimination ability of the discriminator between SR images and HR images in terms of details, and may cause high-frequency detail information to be lost in SR images generated from inputs containing rich details and high-frequency content, thereby compromising visual fidelity.

In other words, we should distinguish the weights of the corresponding loss functions for images with different proportions of high-frequency information in the training samples. Images with a higher proportion of high-frequency information usually contain more detailed information and deserve more attention toward the perceptual loss during super-resolution, while images with more low-frequency information (such as sky backgrounds) require more focus on pixel loss. Based on this, we propose Support GAN(SupGAN), and the contributions of this paper are as follows:

- We design a multi-scale feature-extraction block in the generator network. It features a five-level perceptual hierarchy that captures multi-scale features from LR images while suppressing artifacts.
- We propose a network-training strategy that adjusts learning priorities based on the proportion of high-frequency information in the training images (BHFTM), which includes an edge detection module, a weight-setting module, and a loss-construction module, to dynamically adjust the weights of loss functions in the super-resolution model according to the proportion of high-frequency information in the training samples.
- We employ the four-patch method to better simulate the degradation of complex real-world scenarios.
- We demonstrate that SupGAN produces SR images with better recovery of high-frequency details and improved perceptual metrics while reducing the occurrence of artifacts, resulting in SR images that better meet human perceptual needs (see Figure 1).

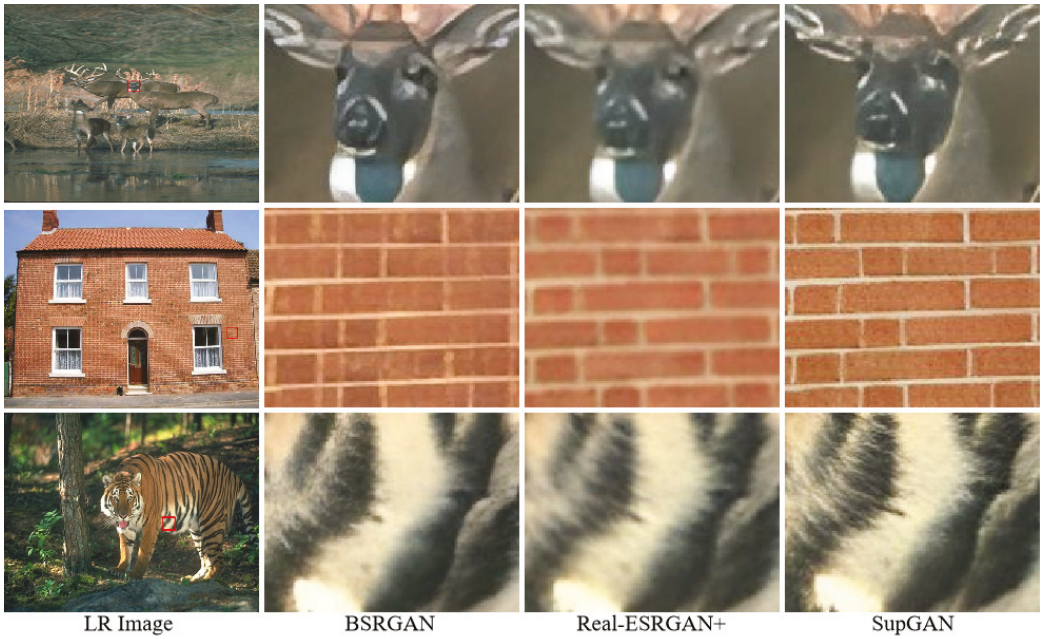


Figure 1. Comparisons of visual quality among BSRGAN, Real-ESRGAN+, and SupGAN on real-life images with $\times 4$ upscaling. The SupGAN can generate naturally finer textures and remove or alleviate annoying artifacts in real-world images. Zoom-in for the best view.

2. Related Work

Single image super-resolution: Over the past few years, deep learning-based approaches have achieved remarkable progress in the field of single-image super-resolution (SISR) [6,12,13]. SRCNN [14] was the first to introduce deep learning into super-resolution, significantly outperforming traditional methods. Dong et al. later proposed FSRCNN [15], which improved on SRCNN by offering a faster and more real-time solution. ESPCN [16] introduced a subpixel convolution layer that extracts features directly from low-resolution input, thus avoiding convolutions in high-resolution images and reducing computational cost. Generative Adversarial Networks (GANs) have also been widely adopted in SISR tasks, utilizing adversarial training between a generator and a discriminator to produce more realistic high-resolution outputs [7,8,10]. More recently, researchers have begun exploring the powerful modeling capabilities of transformers [17] in computer vision. For example, Yang et al. present TTSR [18], which formulates LR and HR images as queries and keys within the transformer architecture to promote joint feature learning between LR and HR representations.

Perception-driven approaches: Image super-resolution methods that optimize for peak signal-to-noise ratio (PSNR) often produce overly smooth high-resolution images that lack fine high-frequency details and fail to align with human perceptual quality. To address this limitation, perception-driven approaches have gained significant attention in recent years. RCAN [19] enhances perceptual quality by employing residual blocks along with channel attention mechanisms within its super-resolution architecture. To capture multi-scale information and improve feature discrimination, RFB-ESRGAN [20] integrates a Receptive Field Block (RFB) [21] into its design and achieved top performance in the NTIRE 2020 Perceptual Extreme Super-Resolution Challenge.

Loss function: Traditional super-resolution methods commonly use mean squared error (MSE) as the loss function [14]. While effective for optimizing pixel-wise accuracy, MSE tends to produce overly smooth high-resolution outputs that lack high-frequency details. Although the use of L1 loss has shown improvements over MSE [22], it still fails to adequately capture the perceptual quality of the generated images. To address this issue, Li

Fei-Fei et al. introduced a perceptual loss function [23], based on the concept of perceptual similarity [24]. This loss evaluates the perceptual difference between the reconstructed and ground-truth images by computing feature distances within a pre-trained deep neural network. SRGAN [7] introduced adversarial loss, employing a discriminator network to guide the generator towards producing images that lie on the natural image manifold. To further enhance texture realism, Sajjadi et al. proposed a texture matching loss [25], which compares the texture statistics between generated and real images. Contextual loss [26] was also proposed to better capture structural and content-level similarities between images. Since a single loss function often cannot fully account for the diverse aspects of image quality in super-resolution tasks, many modern approaches adopt a combination of multiple loss terms. A commonly used composite loss includes L1 loss, perceptual loss, and adversarial loss [7,8,10].

Degradation models: Deep learning-based super-resolution methods often rely on classical degradation models [27,28], which typically consist of a sequence of operations, including blurring, downsampling, and noise addition. However, the performance of these methods can degrade significantly when real-world degradation deviates from these assumed models. To handle this situation, some approaches use implicit degradation modeling, learning abstract representations of degradation types in the feature space rather than explicitly modeling them in the pixel space [29–31]. BSRGAN [32] proposes a more practical and diverse degradation model by randomly shuffling different degradation operations, including blur, downsampling, and noise, while Real-ESRGAN [10] further improves on this by introducing a higher-order degradation process designed to more accurately reflect the complexity of real-world image corruptions.

Evaluation metrics: Super-resolution methods based on deep convolutional neural networks are typically evaluated using two categories of metrics: distortion-based metrics (e.g., PSNR, SSIM, IFC, VIF [33–35]) and perceptual quality metrics. The latter includes human subjective assessments and no-reference image quality indicators such as the Ma score [36], NIQE [37], BRISQUE [38], and the perceptual index (PI) [39,40]. Yochai et al. [39] highlighted the inherent trade-off between distortion accuracy and perceptual quality, where methods achieving high perceptual quality often score lower on traditional distortion metrics such as PSNR and SSIM. In this work, we adopt NIQE as our primary evaluation metric. As a no-reference metric, NIQE assesses the perceptual quality of reconstructed images without requiring access to ground truth images [41].

3. Proposed Methods

To enhance perceptual quality and reduce artifacts in single-image super-resolution (SISR), we introduce Support GAN (SupGAN). This section details three key components: the generator architecture with its multi-scale feature-extraction block (MFEB), enhancements to high-order degradation modeling through an improved four-patch strategy, and the balanced high-frequency training method (BHFTM) for network optimization.

3.1. Generator Architecture

Studies demonstrate that convolutional filters with varying kernel sizes acquire distinct feature extraction capabilities, where larger kernels capture broader structural patterns leading to performance gains [14]. Our MFEB module employs an inception-inspired design [42], with five parallel convolutional pathways for hierarchical feature extraction from low-resolution inputs, as illustrated in Figure 2.

The pathway configuration is as follows:(1) micro-feature pathway (k1-n64-s1) preserves fine details through minimal receptive fields; (2) fine-feature pathway (k1-n32-s1 $\rightarrow$ k3-n64-s1) combines small kernels for subtle pattern retention; (3) mid-range pathway

(k1-n32-s1 → k3-n48-s1 → k3-n64-s1) employs medium kernels for intermediate features; (4) broad-context pathway (k1-n32-s1 → k3-n40-s1 → k3-n48-s1 → k5-n64-s1) utilizes larger kernels for extensive feature capture; (5) global-context pathway (k1-n32-s1 → k3-n40-s1 → k3-n48-s1 → k5-n56-s1 → k5-n64-s1) integrates multiple kernel sizes for comprehensive analysis.

All pathway outputs are processed through SiLU activation functions before undergoing pixel-shuffle operations to generate multi-scale feature maps F_1–F_3. The complete transformation is mathematically represented as

$$F_i = [SiLU(Convs_i(x_{sub}))] \uparrow_s, i \in \{1, 2, 3\}. \quad (1)$$

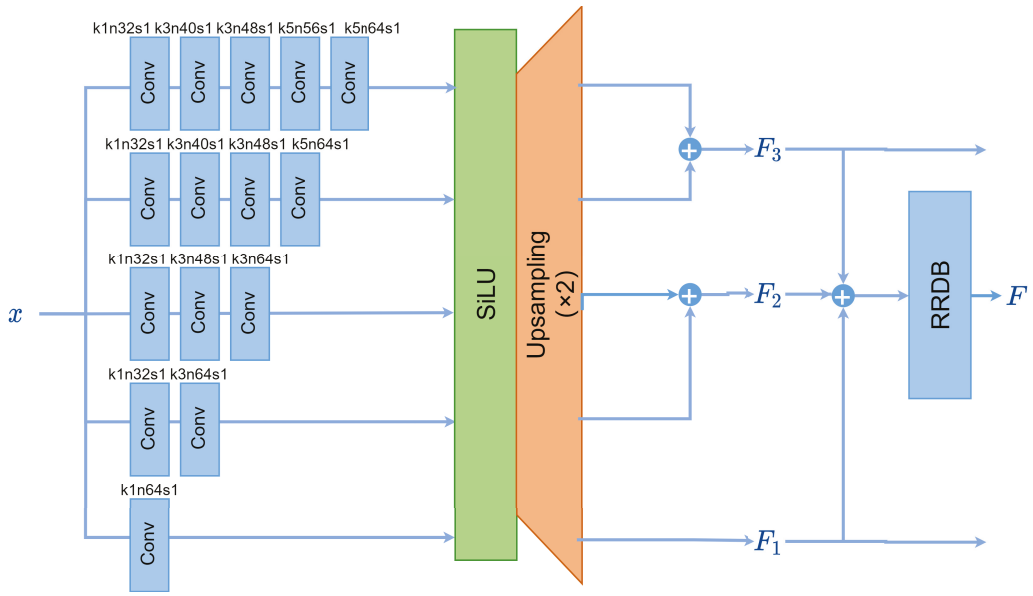


Figure 2. Design of a MFEB in the generator network.

Finally, these features are mixed to obtain a richer representation of features.

3.2. Four-Patch Degradation Model

The four-patch method is proposed to improve the downsampling operation in the Real-ESRGAN high-order degradation modeling process. The main reason for this improvement is that the interpolation method used during the resize stage is singular, selecting one among bicubic, area, and bilinear interpolation methods. However, in the field of super-resolution (SR), it cannot be definitively stated which interpolation method is the best. Each interpolation method represents a distribution of a specific type of low-resolution (LR) image in the real world and cannot represent all types of LR images. Existing SR approaches in the field typically apply a single interpolation method to a single image during the process of handling degraded images, and then train the network based on it. However, this approach deviates significantly from real-world scenarios.

Therefore, during the resize process, it is necessary to calculate a single image using multiple interpolation algorithms. The specific approach is to divide the image into four patches, where consecutive pairs of patches are processed using different interpolation methods. The interpolation method is distributed as Figure 3.

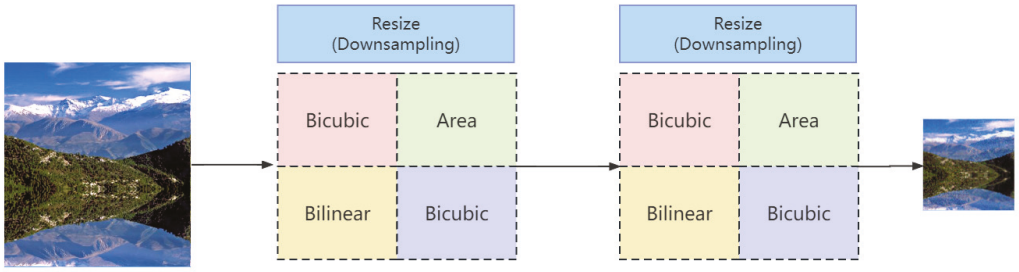


Figure 3. Design of four-patch degradation model.

3.3. Network Training Method

The composition of our method is illustrated in Figure 4. It includes an edge detection module, a weight setting module, and a loss construction module.

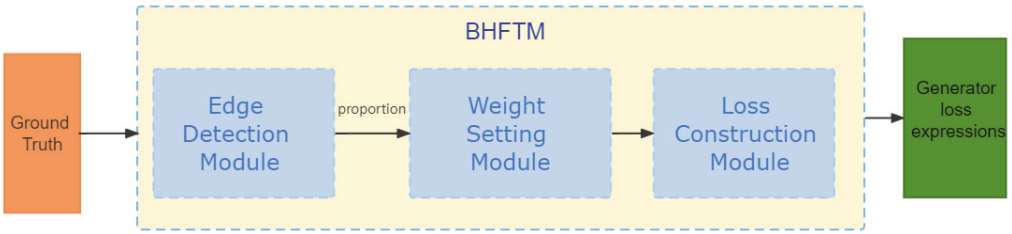


Figure 4. Design of BHFTM.

Edge detection module: To differentiate the proportion of high-frequency information in images, we use edge detection algorithms to preprocess the images, converting digital images into images composed of lines and curves, providing shape and boundary information of objects. We compared several commonly used edge detection algorithms and used the Canny operator to detect image edges. The detection effect of the Canny operator is shown in Figure 5.

The Canny operator [43] is an edge detection algorithm based on Gaussian filtering and non-maximum suppression. It has good edge detection performance and is suitable for detecting fine-edge parts. Compared with other first-order differential operators (such as Roberts operator [44], Sobel operator [45], and Prewitt operator [46]), the Canny operator has stronger noise reduction ability.

Weight-setting module: Image samples with a lower proportion of high-frequency content contain less effective information, and, during training, these samples should be trained to produce high-definition images with equally limited information. To achieve this, it is necessary to reduce the proportion of the perceptual loss and adversarial loss and emphasize the role of the L1 loss. Therefore, the L1 loss weight is increased for samples with a lower proportion of high-frequency content, and decreased for those with a higher proportion. The main process of the BHFTM, which adjusts the weights of the L1 loss for each sample based on the proportion of high-frequency information obtained by the edge detection module, is described in Algorithm 1.

Loss-construction module: The generator's loss function combines pixel loss L_1, perceptual loss L_{Percep}, and adversarial loss L_{GAN} into a single unified formula. The overall loss function of the generator can be expressed as

$$\begin{aligned} L_G =& \alpha L_1 + \beta L_{Percep} + \gamma L_{GAN} \\ =& \alpha \mathbb{E}_{x_i} \|G(x_i) - y\|_1 + \beta \mathbb{E}_{x_i} \|VGG(G(x_i)) - VGG(y)\|_1 \\ &+ \gamma \left(-\mathbb{E}_{x_{hr}} [\log(1 - D(x_{hr}, x_{sr}))] - \mathbb{E}_{x_{sr}} [\log(D(x_{sr}, x_{hr}))] \right) \end{aligned} \tag{2}$$

where L_1 is the 1-norm distance between the recovered image $G(x_i)$ and HR image y, evaluating the average degree of approximation of I^{SR} and I^{HR} on pixels. L_{Percep} is gained by introducing a fine-tuned VGG19 network to calculate the 1-norm distance between the recovered image $G(x_i)$ and high-level features of y, assessing the perceptual similarity between I^{SR} and I^{HR} from a human perspective. The L_{GAN} framework leverages the discriminator's advanced perceptual discrimination capability to differentiate between super-resolved and high-resolution images, thereby enhancing the reconstruction of sharp edges and fine texture details. The loss weights α, β and γ are determined by the weight-setting module:

$$\alpha = \begin{cases} 0 & \text{if } p > 0.2 \\ (1-p) \cdot 10 & \text{otherwise} \end{cases}, \quad \beta = 1.0, \quad \gamma = 0.1. \tag{3}$$

Algorithm 1 Edge detection and weight setting.

Input: GT image set $\mathcal{X}$.
Output: Weight of L1 loss, Perceptual loss and Adversarial loss α, β, γ.

1: **for all** I_{GT} such that $I_{GT} \in \mathcal{X}$ **do**
2: generate proportion of high-frequency information p through edge detection module from I_{GT}.
3: Get α, β, γ through weight setting module by p
4: **if** $p > 0.2$ **then**
5: $\alpha = 0, \beta = 1.0, \gamma = 0.1$
6: **else**
7: $\alpha = (1-p) * 10, \beta = 1.0, \gamma = 0.1$
8: **end if**
9: **end forreturn** α, β, γ.



Figure 5. The detection effect of the Canny operator.

4. Experiments

4.1. Training Details

In our experimental setup, we utilize a ×4 scaling ratio to generate low-resolution (LR) images from their high-resolution (HR) counterparts. The HR images are initially cropped into 400 × 400 patches, which are subsequently degraded through our enhanced four-patch-augmented high-order degradation approach [10] to produce LR images at 100 × 100 resolution. The cropped HR and LR patches maintain dimensions of 256 × 256 and 64 × 64 respectively. For model training, we employ a distributed batch configuration of 12 × 2, indicating 12 samples per GPU across two parallel GPUs.

The training protocol consists of two distinct phases: generator pre-training followed by adversarial training. During the initial phase, the generator undergoes exclusive optimization using L1 loss with a learning rate of 2×10^{-4} over 400,000 iterations. This pre-trained generator then serves as the foundation for subsequent adversarial training, where it collaborates with the discriminator under a composite objective function incorporating L1 loss, perceptual loss, and adversarial loss. The BHFTM algorithm dynamically balances these loss components throughout training. Both networks maintain a constant learning rate of 1×10^{-4} throughout the 280,000-iteration adversarial phase. This staged approach yields two benefits: (1) it prevents generator convergence to inferior local optima, producing superior visual results, and (2) it refines the discriminator's capacity to discern textural details, thereby elevating the quality of super-resolved images during adversarial training.

Optimization is performed using the Adam algorithm [47] with momentum coefficients $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The generator and discriminator undergo alternating updates until model convergence. All implementations were developed using the PyTorch framework and were executed on NVIDIA GeForce RTX 3090 GPU hardware.

4.2. Data

We trained our model on the DIV2K [48], Flickr2K [13] and the OutdoorSceneTraining(OST) [49] dataset, whose abundant high-resolution images with complex textures facilitate the synthesis of more natural super-resolved results [8]. Experimental validation was conducted on several widely recognized benchmarks, such as OST300 [49], BSDS100 and BSD200 [50], 2020track1 [51], Urban100 [52], and RealWorld38 [31], ensuring a thorough comparison with existing methods.

4.3. Qualitative Results

To assess the effectiveness of our approach, we compared SupGAN with several leading SR techniques, including Real-ESRGAN, BSRGAN, PDM_SR, SwinIR, LDL, and RealSR, using the Natural Image Quality Evaluator (NIQE) as the primary evaluation metric. Table 1 summarizes their performance across six benchmark datasets, where a lower NIQE score indicates superior perceptual quality.

Table 1. NIQE scores on diverse testing datasets: the lower, the better. Colors *Red, Green, Blue* indicate the best 1st, 2nd, and 3rd NIQE results among models on each dataset row. The calculation method of NIQE is derived from the basic SR package of PyTorch 1.11.0+cu113.

	Bicubic	BSRGAN	PDM_SR	SwinIR	LDL	RealSR	Real-ESRGAN+	SupGAN
OST300	7.600	3.309	4.319	2.921	**2.817**	3.521	**2.806**	**2.633**
RealWorld38	7.701	4.677	4.939	**4.452**	**4.376**	4.606	4.538	**4.302**
BSDS100	7.548	3.865	4.785	**3.623**	3.788	**3.773**	3.902	**3.257**
BSDS200	7.576	4.081	5.029	**3.833**	**3.910**	3.968	4.124	**3.536**
urban100	7.235	4.258	4.984	4.080	**3.637**	**4.024**	4.193	**3.820**
2020track1	7.596	3.783	4.101	**3.622**	**3.181**	3.790	3.820	**3.372**

Visual comparisons, along with corresponding NIQE measurements, are provided in Figure 6 and Table 1. The results demonstrate that SupGAN consistently generates sharper and more natural reconstructions with fewer artifacts compared to existing methods. For example, in reconstructing fine textures such as animal fur (see 0934) and architectural details (see 157036), SupGAN produces more realistic outputs than BSRGAN and SwinIR, which often exhibit distorted textures and noise. Meanwhile, RealSR and Real-ESRGAN+ struggle to recover high-frequency details effectively.

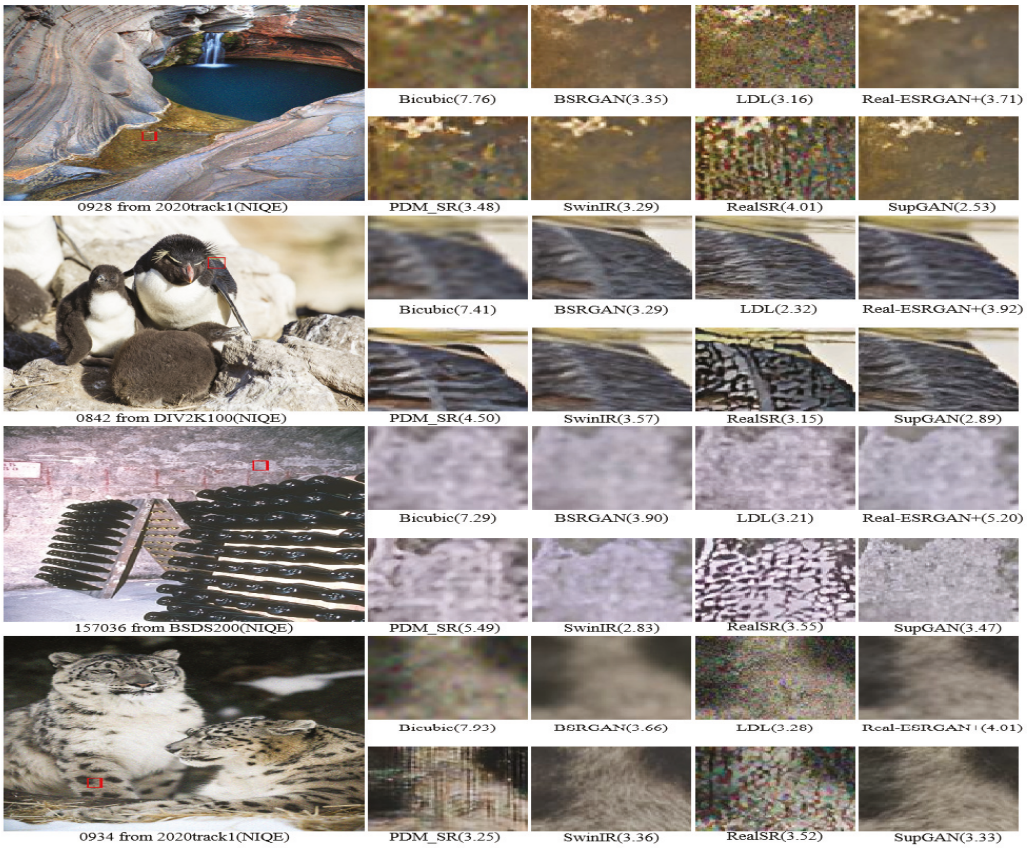


Figure 6. Qualitative results of SupGAN. SupGAN produces more subtle textures and clearer structures, animal textures, and building structures, and also fewer unpleasant artifacts. Zoom in for best view.

Furthermore, SupGAN enhances edge clarity and structural sharpness (see 0842), whereas competing methods either oversmooth details (Real-ESRGAN+ and SwinIR) or fail to restore sufficient high-frequency components (PDM_SR). Notably, while other GAN-based approaches like LDL and RealSR occasionally introduce unnatural artifacts, SupGAN suppresses such distortions, yielding cleaner and more visually coherent results (see 0928).

Although SupGAN does not achieve the lowest NIQE scores in all cases, we emphasize that human perceptual quality should take precedence in SR evaluation, as current quantitative metrics do not fully capture real-world visual fidelity. Our method prioritizes naturalness and detail preservation, aligning better with human judgment than metrics alone.

4.4. Ablation Study

To validate the effectiveness of the BHFTM and four-patch methods, we applied them to the Real-ESRGAN and BSRGAN models and compared them to the original approaches. We have presented some representative qualitative results with NIQE in Table 2. It can be observed that the improved Real-ESRGAN and BSRGAN models with BHFTM and

four-patch methods have shown significant improvements in terms of the NIQE metric. This validates the effectiveness of the BHFTM and four-patch methods.

Table 2. NIQE scores on diverse testing datasets: the lower, the better. Colors *Red* indicate the best NIQE results among models on each dataset row. The calculation method of NIQE is derived from the basic SR package of PyTorch 1.11.0+cu113.

	RealEsrgan+		SupGAN		BSRGAN	
	Origin	Origin+Sup	Origin	Origin+Sup	Origin	Origin+Sup
OST300	2.806	**2.685**	2.637	**2.633**	3.309	**3.151**
2020track1	3.82	**3.631**	3.489	**3.372**	3.783	**3.650**
RealWorld38	4.538	**4.386**	4.339	**4.302**	4.677	**4.201**
BSDS100	3.899	**3.412**	3.302	**3.257**	3.863	**3.745**
BSDS200	4.123	**3.72**	3.583	**3.536**	4.076	**3.841**
urban100	4.193	**3.913**	3.985	**3.820**	4.258	**3.948**

4.5. Efficiency Analysis

Table 3 shows the inference speeds of the original model, the loss-GAN with BHFTM, and the complete Sup-GAN. It can be seen that the introduction of these modules has a negligible impact on speed. Figure 7 and Figure 8 shows the differences in visual effects between each model.

Table 3. Comparison of inference speeds for the original model, loss-GAN with BHFTM, and Sup-GAN, indicating minimal overhead introduced by added modules.

	Real-ESRGAN	Loss-Real-ESRGAN	Sup-Real-ESRGAN	BSRGAN	Loss-BSRGAN	Sup-BSRGAN
OST300	11 m 24 s	11 m 51 s	**11 m 17 s**	**10 m 10 s**	10 m 56 s	10 m 54 s
2020track 1	1 m 52 s	**1 m 42 s**	2 m 10 s	**2 m 4 s**	2 m 11 s	2 m 14 s
RealWorld38	**22 s**	23 s	24 s	17 s	**13 s**	**13 s**
DIV2K 100	**2 m**	2 m 32 s	2 m 34 s	**2 m 11 s**	2 m 17 s	2 m 15 s
BSDS100	**1 m 45 s**	2 m 2 s	2 m	**1 m 51 s**	2 m 4 s	2 m 7 s
BSDS200	3 m 28 s	3 m 6 s	**3 m 5 s**	**3 m 55 s**	4 m 8 s	4 m 10 s
Urban100	8 m 53 s	**8 m 25 s**	8 m 54 s	**11 m 8 s**	11 m 10 s	11 m 12 s

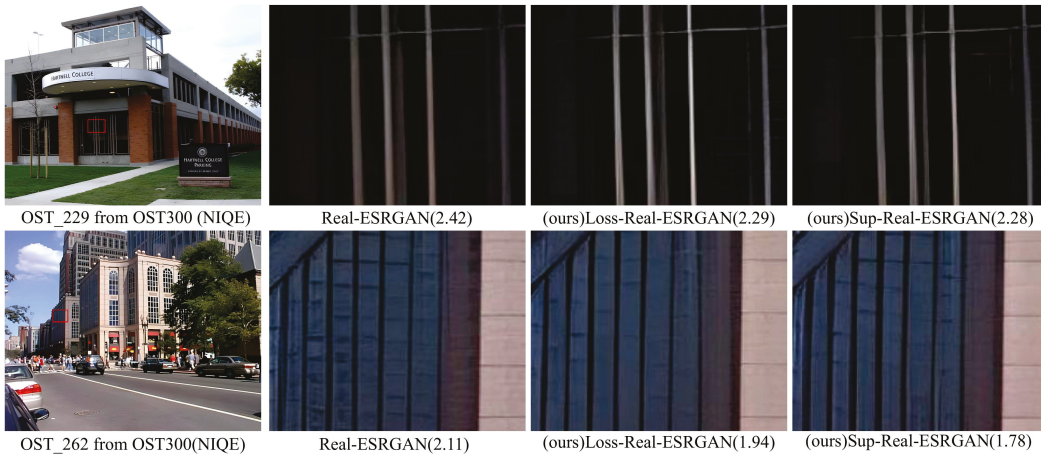


Figure 7. Compared to the original Real-ESRGAN, both Sup-Real-ESRGAN and Loss-Real-ESRGAN restored clearer door frame structures in the first image and eliminated unnecessary distorted horizontal stripes in the second image. The door-frame tones generated by Sup-Real-ESRGAN are softer, closer to the style of the original image, and the door-frame structure is clearer.

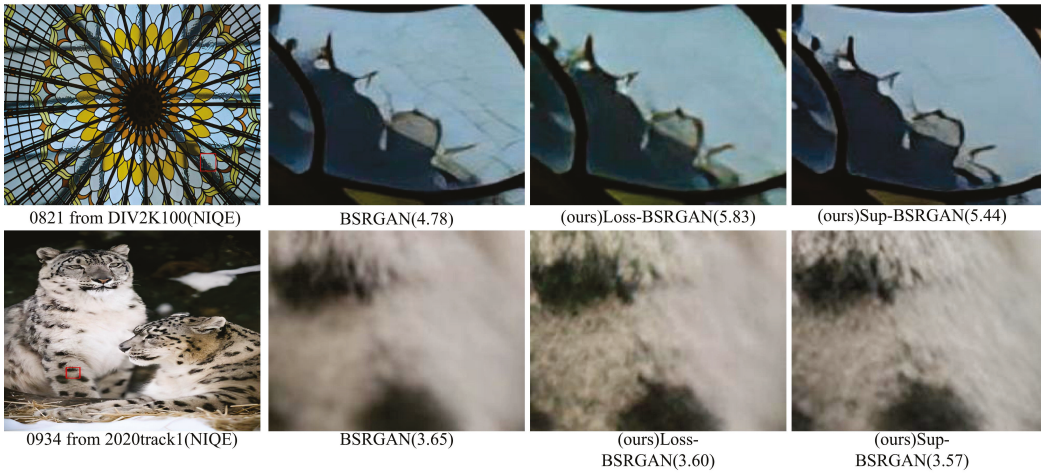


Figure 8. Both Sup-BSRGAN and Loss-BSRGAN, compared to the original BSRGAN, eliminate unnecessary cracks in the first row image and restore clearer animal fur textures in the second row image. The animal fur colors generated by Sup-BSRGAN are lighter than those generated by Loss-BSRGAN, more accurately reflecting the white fur of snow leopards in the real world.

5. Conclusions

We propose a SupGAN that achieves superior perceptual quality through profound NIQE tests. The SupGAN model uses a multi-scale feature extraction and fusion generator for a more comprehensive perception of image features. The four-patch method used can better simulate the complex degradation in reality. SupGAN can produce more natural and detailed textures while suppressing unpleasant artifacts. In addition, it can dynamically adjust the weights of the loss function based on the proportion of high- and low-frequency information in the image.

Author Contributions: T.W.: Project administration, Funding; S.X.g: Writing—review & editing, Funding; Q.C.: Writing—original draft; H.L.: Software, Writing—review & editing; W.C.: Validation, Writing—review & editing; H.T.: Resources, Writing—original draft. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Tencent Co.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: This research is funded by Tencent, and the code-related technologies belong to Tencent. Before allowed by company, we cannot disclose them.

Acknowledgments: We would like to thank Liang Yu and Jiajun Liu, who took over the work of Huaizheng Liu and Weijun Cao, respectively, contributed to revising the manuscript, 280 and thus meet the criteria for authorship with their significant contributions.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhang, L.; Wu, X. An edge-guided image interpolation algorithm via directional filtering and data fusion. *IEEE Trans. Image Process.* **2006**, *15*, 2226–2238. [CrossRef] [PubMed]
2. Duchon, C.E. Lanczos filtering in one and two dimensions. *J. Appl. Meteorol. Climatol.* **1979**, *18*, 1016–1022. [CrossRef]
3. Dai, S.; Han, M.; Xu, W.; Wu, Y.; Gong, Y.; Katsaggelos, A.K. Softcuts: A soft edge smoothness prior for color image super-resolution. *IEEE Trans. Image Process.* **2009**, *18*, 969–981. [CrossRef] [PubMed]
4. Sun, J.; Xu, Z.; Shum, H.-Y. Image super-resolution using gradient profile prior. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.

5. Yan, Q.; Xu, Y.; Yang, X.; Nguyen, T.Q. Single image super-resolution based on gradient profile sharpness. *IEEE Trans. Image Process.* **2015**, *24*, 3187–3202. [PubMed]
6. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
7. Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.; Tejani, A.; Totz, J.; Wang, Z.; et al. Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 4681–4690.
8. Wang, X.; Yu, K.; Wu, S.; Gu, J.; Liu, Y.; Dong, C.; Qiao, Y.; Loy, C.C. ESRGAN: Enhanced super-resolution generative adversarial networks. In Proceedings of the European Conference on Computer Vision Workshops, Munich, Germany, 8–14 September 2018; pp. 63–79.
9. Jolicoeur-Martineau, A. The relativistic discriminator: A key element missing from standard gan. *arXiv* **2018**, arXiv:1807.00734. [CrossRef]
10. Wang, X.; Xie, L.; Dong, C.; Shan, Y. Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data. In Proceedings of the ICCVW, Montreal, BC, Canada, 11–17 October 2021; pp. 1905–1914.
11. Kim, J.; Lee, J.K.; Lee, K.M. Accurate image super-resolution using very deep convolutional networks. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1646–1654.
12. Lai, W.-S.; Huang, J.-B.; Ahuja, N.; Yang, M. Deep laplacian pyramid networks for fast and accurate super-resolution. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 5835–5843.
13. Timofte, R.; Agustsson, E.; Van Gool, L.; Yang, M.H.; Zhang, L.; Lim, B.; Son, S.; Kim, H.; Nah, S.; Lee, K.M.; et al. Ntire 2017 challenge on single image super-resolution: Methods and results. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Honolulu, HI, USA, 21–26 July 2017; pp. 1110–1121.
14. Dong, C.; Loy, C.C.; He, K.; Tang, X. Learning a deep convolutional network for image super-resolution. In Proceedings of the ECCV, Zurich, Switzerland, 6–12 September 2014; pp. 184–199.
15. Dong, C.; Loy, C.C.; Tang, X. Accelerating the Super-Resolution Convolutional Neural Network. In Proceedings of the ECCV, Amsterdam, The Netherlands, 11–14 October 2016; pp. 391–407.
16. Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A.P.; Bishop, R.; Rueckert, D.; Wang, Z. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 1874–1883.
17. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
18. Yang, F.; Yang, H.; Fu, J.; Lu, H.; Guo, B. Learning texture transformer network for image super-resolution. In Proceedings of the CVPR, Seattle, WA, USA, 13–19 June 2020; pp. 5791–5800.
19. Zhang, Y.; Li, K.; Li, K.; Wang, L.; Zhong, B.; Fu, Y. Image super-resolution using very deep residual channel attention networks. In Proceedings of the ECCV, Munich, Germany, 8–14 September 2018; pp. 286–301.
20. Shang, T.; Dai, Q.; Zhu, S.; Yang, T.; Guo, Y. Perceptual Extreme Super Resolution Network with Receptive Field Block. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 14–19 June 2020; pp. 1778–1787.
21. Liu, S.; Huang, D. Receptive field block net for accurate and fast object detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 385–400.
22. Zhao, H.; Gallo, O.; Frosio, I.; Kautz, J. Loss functions for neural networks for image processing. *arXiv* **2015**, arXiv:1511.08861.
23. Johnson, J.; Alahi, A.; Li, F.-F. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*; Springer International Publishing: Cham Switzerland, 2016; pp. 694–711.
24. Bruna, J.; Sprechmann, P.; LeCun, Y. Super-resolution with deep convolutional sufficient statistics. *arXiv* **2015**, arXiv:1511.05666.
25. Sajjadi, M.S.M.; Scholkopf, B.; Hirsch, M. EnhanceNet: Single image super-resolution through automated texture synthesis. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 4501–4510.
26. 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), Munich, Germany, 8–14 September 2018; pp. 800–815.

27. Elad, M.; Feuer, A. Restoration of a single super-resolution image from several blurred, noisy, and undersampled measured images. *IEEE Trans. Image Process.* **1997**, *6*, 1646–1658. [CrossRef] [PubMed]
28. Liu, C.; Sun, D. On bayesian adaptive video super resolution. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *36*, 346–360. [CrossRef] [PubMed]
29. Yuan, Y.; Liu, S.; Zhang, J.; Zhang, Y.; Dong, C.; Lin, L. Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Salt Lake City, UT, USA, 18–23 June 2018.
30. Fritsche, M.; Gu, S.; Timofte, R. Frequency separation for real-world super-resolution. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Republic of Korea, 27–28 October 2019; pp. 3599–3608.
31. Wang, L.; Wang, Y.; Dong, X.; Xu, Q.; Yang, J.; An, W.; Guo, Y. Unsupervised Degradation Representation Learning for Blind Super-Resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 10581–10590.
32. Zhang, K.; Liang, J.; Gool, L.V.; Timofte, R. Designing a practical degradation model for deep blind image super-resolution. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 4771–4780.
33. Sheikh, H.R.; Bovik, A.C.; Veciana, G.D. An information fidelity criterion for image quality assessment using natural scene statistics. *IEEE Trans. Image Process.* **2005**, *14*, 2117–2128. [CrossRef] [PubMed]
34. Sheikh, H.R.; Bovik, A.C. Image information and visual quality. In Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, Montreal, QC, Canada, 17–21 May 2004; pp. 709–712.
35. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
36. Ma, C.; Yang, C.Y.; Yang, X.; Yang, M.H. Learning a noreference quality metric for single-image super-resolution. *Comput. Vis. Image Underst.* **2017**, *158*, 1–16. [CrossRef]
37. Mittal, A.; Soundararajan, R.; Bovik, A.C. Making a "completely blind" image quality analyzer. *IEEE Signal Process. Lett.* **2012**, *20*, 209–212. [CrossRef]
38. Mittal, A.; Moorthy, A.K.; Bovik, A.C. No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Process.* **2012**, *21*, 4695–4708. [CrossRef] [PubMed]
39. Blau, Y.; Michaeli, T. The perception-distortion tradeoff. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 6228–6237.
40. Vasu, S.; Madam, N.T.; Rajagopalan, A.N. Analyzing Perception-Distortion Tradeoff Using Enhanced Perceptual Super-Resolution Network. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018; pp. 114–131.
41. Blau, Y.; Mechrez, R.; Timofte, R.; Michaeli, T.; Zelnik-Manor, L. The 2018 PIRM Challenge on Perceptual Image Super-Resolution. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 September 2018; pp. 334–355.
42. Christian, S.; Wei, L.; Yangqing, J.; Pierre, S.; Scott, R.; Dragomir, A.; Dumitru, E.; Vincent, V.; Andrew, R. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
43. Canny, J. A Computational Approach to Edge Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *6*, 679–698.
44. Roberts, L.G. Machine Perception of Three-Dimensional Solids. Optical and Electro-Optical Information Processing. Doctoral's Dissertation, Massachusetts Institute of Technology, Cambridge, MA, USA, 1963.
45. Sobel, I. An isotropic 3×3 image gradient operator. In *Machine Vision for Three-Dimensional Scenes*; Freeman, H., Ed.; Academic Press: San Diego, CA, USA, 1990; pp. 376–379.
46. Prewitt, J.M. Object enhancement and extraction. *Pict. Process. Psychopictorics* **1970**, *10*, 15–19.
47. Kingma, D.P.; Ba, J.L. Adam: A method for stochastic optimization. In Proceedings of the ICLR, San Diego, CA, USA, 7–9 May 2015; pp. 1–15.
48. Agustsson, E.; Timofte, R. Ntire 2017 challenge on single image super-resolution: Dataset and study. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Honolulu, HI, USA, 21–26 July 2017; pp. 1122–1131.
49. Wang, X.; Yu, K.; Dong, C.; Loy, C.C. Recovering realistic texture in image super-resolution by deep spatial feature transform. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 606–615.
50. Martin, D.; Fowlkes, C.; Tal, D.; Malik, J. A database of human segmented naturalimages and its application to evaluating segmentation algorithms and measuringecological statistics. In Proceedings of the Eighth IEEE International Conference on Computer Vision, Vancouver, BC, Canada, 7–14 July 2001.

51. Lugmayr, A.; Danelljan, M.; Timofte, R. Ntire 2020 challenge on real-world image super-resolution: Methods and results. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 494–495.
52. Huang, J.B.; Singh, A.; Ahuja, N. Single image super-resolution from transformedself-exemplars. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

Salp Swarm Algorithm Optimized A* Algorithm and Improved B-Spline Interpolation in Path Planning

Hang Zhou, Tianning Shang, Yongchuan Wang and Long Zuo *

School of Information Engineering, Chang'an University, Xi'an 710064, China; zhouh1214@163.com (H.Z.); shangtianning1101@163.com (T.S.); wangyongchuanchd@163.com (Y.W.)
* Correspondence: zuolong@chd.edu.cn

Abstract: The efficiency and smoothness of path planning algorithms are critical factors influencing their practical applications. A traditional A* algorithm suffers from limitations in search efficiency, path smoothness, and obstacle avoidance. To address these challenges, this paper introduces an improved A* algorithm that integrates the Salp Swarm Algorithm (SSA) for heuristic function optimization and proposes a refined B-spline interpolation method for path smoothing. The first major improvement involves enhancing the A* algorithm by optimizing its heuristic function through the SSA. The heuristic function combines Chebyshev distance, Euclidean distance, and obstacle density, with the SSA adjusting the weight parameters to maximize efficiency. The simulation experimental results demonstrate that this modification reduces the number of searched nodes by more than 78.2% and decreases planning time by over 48.1% compared to traditional A* algorithms. The second key contribution is an improved B-spline interpolation method incorporating a two-stage optimization strategy for smoother and safer paths. A corner avoidance strategy first adjusts control points near sharp turns to prevent collisions, followed by a path obstacle avoidance strategy that fine-tunes control point positions to ensure safe distances from obstacles. The simulation experimental results show that the optimized path increases the minimum obstacle distance by 0.2–0.5 units, improves the average distance by over 43.0%, and reduces path curvature by approximately 61.8%. Comparative evaluations across diverse environments confirm the superiority of the proposed method in computational efficiency, path smoothness, and safety. This study presents an effective and robust solution for path planning in complex scenarios.

Keywords: salp swarm algorithm; A* algorithm; path smoothing; B-spline interpolation

1. Introduction

Path planning [1] is a fundamental technology for robotic navigation and autonomous driving, with its performance directly determining the efficiency and safety of mobile agents. The algorithm must generate optimal and smooth collision-free paths to reduce collision risks. Path planning [2] focuses on discovering an optimal trajectory that avoids obstacles between a starting point and a destination. An ideal path planning algorithm needs to meet two key requirements: first, it must be computationally efficient, capable of generating feasible paths in real time; second, the generated path must be smooth and safe for practical execution.

Path planning algorithms can be broadly classified into two types: global path planning and local path planning. Currently, common global path planning algorithms include graph search-based algorithms, sampling-based algorithms, intelligent optimization algorithms, mathematical model-based algorithms, and hybrid algorithms. Graph

search-based algorithms include Dijkstra's Algorithm [3], A* algorithms [4], and hybrid A* algorithms [5]. Sampling-based algorithms include the Rapidly Exploring Random Tree (RRT) algorithm [6] and RRT* algorithm [7]. Intelligent optimization algorithms include the Genetic Algorithm [8], Ant Colony Optimization (ACO) [9], and Particle Swarm Optimization (PSO) [10]. Mathematical model-based algorithms include the Dynamic Programming (DP) algorithm [11]. Hybrid algorithms [12] combine the advantages of multiple algorithms to improve path quality and search efficiency. The Intelligent Bug algorithm [13] and its variants are currently the most commonly used methods for solving local path planning problems. Among these path planning algorithms, the A* algorithm is recognized as one of the most efficient direct search methods for solving the shortest path in static road networks. Due to its balance of optimality and efficiency [14], it has been widely used.

The A* algorithm combines Dijkstra's comprehensive search with heuristic guidance, greatly reducing unnecessary node expansion and improving search efficiency. However, as more scholars have delved into the A* algorithm, several drawbacks have been identified, such as the generation of non-smooth paths and a decrease in search efficiency when the map size increases.Over the years, both domestic and international scholars have made various improvements to the A* algorithm. Common optimization strategies include improving the heuristic function, enhancing the search neighborhood, and modifying the search strategy. Wu et al. [15] proposed a heuristic function mechanism with directional information, which enhances the algorithm's directional intelligence, helping to reduce computational complexity and time. Zheng et al. [16] introduced the DPIO method, which optimizes the heuristic function to enhance the efficiency and quality of path planning. Zhang et al. [17] optimized the heuristic function and introduced the Jump Point Search strategy, reducing the number of nodes that need to be expanded during the path search process, thus improving the algorithm's efficiency and reducing computational time and memory consumption. Huang et al. [18] proposed a new 5-neighborhood search method, dynamically adjusting the search neighborhood (5-neighborhood, 4-neighborhood, or 3-neighborhood) based on obstacle constraints, thereby improving path smoothness.

Among the aforementioned algorithms, the A* algorithm has shown improvements in both search efficiency and path smoothness. However, these improvements, due to the need for dynamic adaptation, can consume considerable computer resources. Additionally, the Jump Point Search strategy may not effectively handle dense obstacle environments in certain cases. In recent years, with the rapid development of intelligent optimization algorithms, various swarm intelligence optimization algorithms such as the Salp Swarm Algorithm (SSA) [19], Bobcat Optimization Algorithm [20], and Wombat Optimization Algorithm [21] have demonstrated remarkable advantages in the field of path planning due to their excellent global search capabilities and convergence performance. These algorithms, by simulating the intelligent behaviors of biological groups in nature, provide novel approaches for solving complex optimization problems. Therefore, this paper proposes an improved A* algorithm based on the SSA. The algorithm introduces a new A* heuristic function and optimizes its weight coefficients to enhance search efficiency while ensuring path feasibility. Furthermore, this paper also optimizes path smoothing to enhance the safety and smoothness of the final path.

In practical applications, path smoothing is crucial. Traditional path smoothing methods include the Artificial Potential Field [22], Bezier curves [23], and B-spline interpolation [24], and others. Among these, the B-spline interpolation algorithm is the most widely applied. However, traditional methods like B-spline interpolation often struggle to maintain a safe distance from obstacles. To address this, our approach integrates an obstacle avoidance mechanism into the smoothing process, balancing both path smoothness and

safety. We propose a two-stage optimized B-spline interpolation strategy: in the first stage, control point positions are adjusted to avoid collisions with obstacles; in the second stage, global obstacle avoidance is achieved.

In conclusion, this paper focuses on how to optimize the A* algorithm using the SSA to improve the efficiency of path search and proposes an improved B-spline smoothing strategy to guarantee the safety and smoothness of the path. The following chapters will provide a comprehensive explanation of the proposed method, followed by the experimental results and comparative analysis to validate the effectiveness of the proposed approach.

The main structure of this paper is as follows: Section 2 introduces the path planning algorithm, proposing an A* algorithm optimized by the SSA. Section 3 elaborates on the path smoothing algorithm, presenting an improvement of the B-spline interpolation method. The simulation experiments of the two proposed algorithms are demonstrated in Section 4. The conclusions are given in Section 5.

2. Path Planning Algorithm

2.1. The A Algorithm*

The A* algorithm is a heuristic search algorithm proposed by Peter Hart et al. in 1968 [25]. It combines the optimality guarantee of Dijkstra's Algorithm with the efficiency of greedy search. Unlike traditional non-heuristic search methods, the A* algorithm considers both the estimated distance from the current node to the target and the actual cost to reach the node while exploring the path. A combined priority score, $f(n) = g(n) + h(n)$, is computed to intelligently guide the search process. This mechanism enables the A* algorithm to ensure optimal solutions while significantly improving search efficiency, which is why it has been broadly utilized in various fields such as path planning, game AI, and robot navigation.

However, despite its excellent performance in many scenarios, the A* algorithm still has some limitations. First, its computational time can be long, particularly in large maps where the algorithm must expand a large number of nodes and maintain open and closed lists, which increases the computational burden significantly. Second, the performance of the A* algorithm is critically determined by the design of the heuristic function $h(n)$. If the heuristic function is unacceptable or inconsistent, the algorithm may fail to find the optimal solution or experience a significant drop in efficiency.

2.2. Improvement of the A Algorithm*

To address the limitations of the traditional A* algorithm and considering that its performance is highly dependent on the design of the heuristic function $h(n)$, it is important to recognize that the heuristic function not only determines whether the algorithm can find the optimal solution, but also directly affects the search efficiency. The traditional A* algorithm imposes a strict requirement on the heuristic function $h(n)$, which must satisfy $h(n) \leq$ *the actual minimum cost*. This condition guarantees that the algorithm is capable of identifying the shortest possible path; however, it also leads to the expansion of a large number of nodes, resulting in longer computation times. In practical applications, once path planning is completed, the path is often smoothed, a process that may increase the path length. Therefore, during the initial phase of path planning, there is no need to strictly pursue the shortest path, but rather, it is more practical to seek a balance between path quality and computational efficiency.

Based on this understanding, this chapter proposes an improvement strategy: by relaxing the strict requirements on the shortest path, we attempt to construct a new heuristic function, making the A* algorithm no longer focus solely on minimizing the distance. This approach significantly improves the algorithm's computational efficiency while ensuring

the path remains reasonable. This improvement not only reduces the number of nodes expanded by the algorithm and alleviates the computational burden, but also better accommodates the flexible requirements of path planning in real-world applications. To validate this hypothesis, we designed and implemented a new heuristic function and further improved it using the SSA. We refer to this algorithm as the SSA-A*.

2.2.1. Determining the Variables Included in $h(n)$

First, it is necessary to identify the variables in $h(n)$. For an eight-directional search, the traditional A* algorithm's heuristic function $h(n)$ often utilizes either the Chebyshev distance or the Euclidean distance. Among these, the Chebyshev distance is more effective in measuring the number of moves in an eight-directional grid, while the Euclidean distance better represents the actual distance. Therefore, we use both distance metrics simultaneously to enhance the accuracy of the heuristic function $h(n)$ in distance evaluation.

Additionally, considering that when a node is surrounded by many obstacles, the path passing through that node may increase in length due to the need to avoid obstacles, we also incorporate the obstacle density into the heuristic function $h(n)$. This addition enables the search process to possess a certain level of obstacle avoidance capability.

Thus, we modify the heuristic function $h(n)$ as follows (1):

$$\begin{cases} h(n) = \omega_1 * d_{Chebyshev} + \omega_2 * d_{Euclidean} + \omega_3 * \rho_{obstacle} \\ \rho_{Obstacle} = \frac{n_{obstacle}}{8} \end{cases} \quad (1)$$

In this context, $d_{Chebyshev}$ represents the Chebyshev distance from the current node n to the goal node, $d_{Euclidean}$ denotes the Euclidean distance, $\rho_{obstacle}$ represents the obstacle density in the 8-neighborhood of the current node n, and $n_{obstacle}$ refers to the number of obstacles in the 8-neighborhood of node n (as illustrated in Figure 1). $\omega_1, \omega_2, \omega_3$ are weight coefficients.

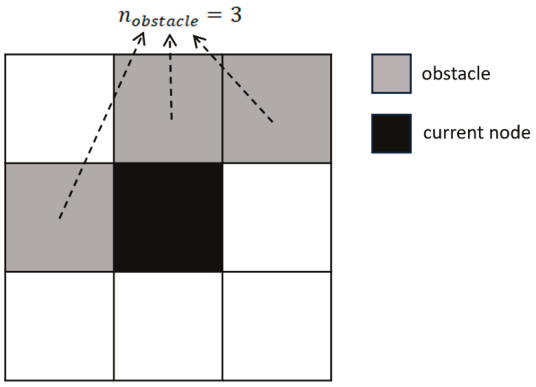


Figure 1. Calculation of obstacle count.

2.2.2. Optimization Using the SSA

To determine the weight coefficients $\omega_1, \omega_2, \omega_3$, an efficient optimization method is required. The combination of these weight coefficients directly impacts the performance of the heuristic function, and therefore, identifying the optimal solution in a complex parameter space is of critical importance.

When selecting an optimization method, we take into account that the optimization problem for the weight coefficients may have multiple local optima. Thus, an algorithm that can effectively prevent being trapped in local optima is required. Moreover, to enhance optimization efficiency, the algorithm should be able to find a near-optimal solution in fewer iterations. Additionally, the algorithm needs to handle nonlinear, multi-modal

optimization problems to accommodate the complexity of the weight coefficients in the heuristic function, and it should be easy to implement and insensitive to initial parameter settings, reducing the complexity of tuning. Based on these requirements, we have chosen the SSA for the optimization of the weight coefficients.

The Salp Swarm Algorithm (SSA) [26], proposed by Seyedali Mirjalili et al. [27] in 2017, is a nature-inspired swarm intelligence algorithm that simulates the foraging behavior of salps (as shown in Figure 2). Salps are marine organisms that search for food through collective cooperation and leader-following behavior. The SSA leverages this behavior to optimize problem solutions. Salps form chain-like swarms in the ocean, and the SSA simulates this behavior by dividing the swarm into leaders and followers. The leader is at the front of the chain and is responsible for guiding the group, while the followers move by following the leader. The SSA simulates the foraging behavior of salp swarms in the ocean, offering strong global search capability and efficiently exploring high-dimensional parameter spaces [28]. Its rapid convergence speed makes it suitable for solving complex optimization problems. Furthermore, the SSA is insensitive to initial parameter settings, simple to implement, and easy to adjust, making it well suited to meet our optimization needs.

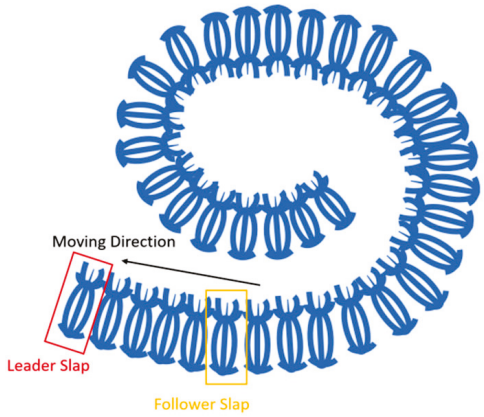


Figure 2. Movement of the salp chain.

Next, the SSA is used to determine the weight coefficients $\omega_1, \omega_2, \omega_3$. First, we initialize the population using (2) and (3).

$$x = \begin{bmatrix} X_1^1 & X_2^1 & \dots & X_d^1 \\ X_1^2 & X_2^2 & \dots & X_d^2 \\ \vdots & \vdots & \ddots & \vdots \\ X_1^n & X_2^n & \dots & X_d^n \end{bmatrix} \tag{2}$$

$$X_j^i = rand * \left(ub_j - lb_j\right) + lb_j, i \in \{1, 2, \dots, n\}, j \in \{1, 2, \dots, n\} \tag{3}$$

X is an $n * d$ matrix, with n representing the population size and d representing the dimensionality of the problem. The terms ub and lb denote the upper and lower bounds of the problem, respectively, while $rand$ is a randomly generated number between 0 and 1.

To enhance the efficiency of the A* algorithm, we define the fitness function as the number of searched nodes, where a lower number of searched nodes indicates a higher fitness score. Each individual's fitness is ranked, with the highest fitness individual designated as the leader and the remaining individuals as followers. Subsequently, both the

leader and followers update their positions according to the predefined formulas. The leader's position is updated using (4) and (5).

$$X_j^i = \begin{cases} F_j + c_1\left[\left(ub_j - lb_j\right) * c_2 + lb_j\right], c_3 \leq 0.5 \\ F_j - c_1\left[\left(ub_j - lb_j\right) * c_2 + lb_j\right], c_3 > 0.5 \end{cases} \tag{4}$$

$$c_1 = 2e^{-\left(\frac{4t}{T}\right)^2} \tag{5}$$

The followers update their positions using (6).

$$X_j^i = \frac{1}{2}\left(X_j^i + X_j^{i-1}\right) \tag{6}$$

After each update, the fitness of each individual is re-evaluated, and the optimization process iterates until a predefined number of iterations is reached or a convergence criterion is met. Finally, the optimal weight parameters are selected. Using these optimal weight parameters, the heuristic function $h(n)$ is determined and subsequently employed for path planning.

The pseudo-code for the optimization of the A* algorithm using the SSA is shown in Algorithm 1 below.

Algorithm 1: Improved A* Algorithm with the SSA Optimization

Input: Grid map G, start point S, end point E, weight bounds Ω_{bounds}, population size N, maximum iterations T
Output: Optimal path P, visited nodes list V, optimal weight Ω^*
BEGIN:
  Initialization:
    Define 8-neighborhood direction set
      $D = \{(-1,0),(1,0),(0,-1),(0,1),(-1,-1),(-1,1),(1,-1),(1,1)\}$
    Initialize SSA population $P = \{\Omega_i \mid \Omega_i \sim Uniform(\Omega_{bounds}), i = 1,2,\ldots,N\}$
    Initialize optimal weight $\Omega^* \leftarrow argmin_{\Omega \in P} Fitness(\Omega)$
    Initialize optimal fitness $F^* \leftarrow Fitness(\Omega^*)$
    Initialize iteration counter $t \leftarrow 0$
//SSA Optimization Process:
  While $t < T$:
    For each $\Omega_i \in P$:
      If $Random() < 0.5$:
        $\Omega_i' \leftarrow \Omega^* + Uniform(-0.3, 0.3)^3$
      Else:
        $\Omega_i' \leftarrow \frac{\Omega_i + \Omega^*}{2}$
      Clip Ω_i to Ω_{bounds}
      $Fi' \leftarrow Fitness\left(\Omega_i'\right)$
      If $F_i' < F^*$:
        $\Omega^* \leftarrow \Omega_i'$
        $F^* \leftarrow F_i'$
        $P \leftarrow \left\{\Omega_i'\right\}$
    Record current optimal fitness F^*
    $t \leftarrow t + 1$

Algorithm 1: *Cont.*

//A* Path Search:Use optimal weight $\Omega*$ to execute A* algorithm:
Initialize open list $O \leftarrow \{(f(S), 0, S)\}$, parent map $C \leftarrow \{\}$, cost map $G \leftarrow \{S : 0\}$, and visited nodes list $V \leftarrow []$
While O is not empty:
    Pop $(f, g, Q) \leftarrow HeapPop(O)$
    If $Q = E$:
        $P \leftarrow ReconstructPath(C, S, E)$
        Return P, V
    Add Q to V
    For each $(dx, dy) \in D$:
        $Q' \leftarrow (Q_x + dx, Q_y + dy)$
        If Q' is out of bounds or $G[Q'] = 1$:
            Continue
        $c \leftarrow DiagonalDistance$ if $\mid dx \mid + \mid dy \mid = 2$ *else* 1
        $g' \leftarrow g + c$
        If $Q' \notin G$ *or* $g' < G[Q']$:
            $G[Q'] \leftarrow g$
            $C[Q'] \leftarrow Q$
            $h \leftarrow Heuristic(Q', E, \Omega^*, G)$
            $f' \leftarrow g' + h$
            Push (f', g', Q') into O
Return $None, V$
END

Our work, compared with recent SSA-based path planning studies, demonstrates distinct focuses and innovations. Our proposed algorithm employs SSA to optimize the heuristic function weights of the A* algorithm, dynamically adjusting the weights by incorporating Chebyshev distance, Euclidean distance, and obstacle density to enhance search efficiency. Guo et al. [29] improved the SSA for AUV path planning by introducing the Levy flight strategy and a nonlinear convergence factor to boost the algorithm's global search capability and convergence speed. Meanwhile, Mohammad AlShabi et al. [30] applied SSA to ground vehicle path planning, exploring a new application scenario for the algorithm.

3. Path Smoothing Algorithm

3.1. B-Spline Interpolation Curve

B-spline is a mathematical method commonly used in computer graphics and data fitting. It consists of a series of control points, knot vectors, and a set of basis functions. The control points determine the shape of the spline curve, allowing shape adjustments by repositioning them. The knot vector defines the distribution of the spline curve in parameter space and determines the influence range of each control point on the spline curve. The algorithm utilizes a set of basis functions to construct the spline curve. By weighting the control points with the corresponding basis functions, a smooth interpolated curve can be obtained. The basis functions are computed using a recursive formula, defined as (7) and (8).

$$k = 0, B_{i,0}(t) = \begin{cases} 1, & t_i \le t \le t_{i+1} \\ 0, & Otherwise \end{cases} \tag{7}$$

$$k > 1, B_{i,k}(t) = \frac{t - t_i}{t_{i+k} - t_i} B_{i,k-1}(t) + \frac{t_{i+k+1} - t}{t_{i+k+1} - t_{i+1}} B_{i+1,k-1}(t) \tag{8}$$

The mathematical formula of the B-spline curve is typically expressed as (9).

$$P(t) = \sum_{i=0}^{n} P_i B_{i,p}(t) \tag{9}$$

where $C(t)$ is the curve, P_i are the control points, $B_{i,p}(t)$ is the basis function of the i-th control point with degree p, and t is the parameter.

3.2. Control Point Adjustment

The B-spline interpolation method generates a smooth path based on the number and position of control points. However, this method lacks inherent obstacle avoidance capabilities. If control points are not properly selected, the planned path might not stay safely away from obstacles, which could cause collisions [31].

Therefore, the path nodes generated by the SSA-A* cannot be directly used as control points for B-spline smoothing. Instead, these nodes must be adjusted in terms of quantity and position before serving as control points for the B-spline smoothing process. This adjustment ensures that the generated smooth path avoids obstacles effectively.

To achieve appropriate control point adjustment, we propose a two-stage optimization strategy: corner avoidance and path obstacle avoidance.

The corner avoidance strategy adjusts the number and position of control points at path corners based on the spatial relationships of path nodes, enabling the path to navigate turns without colliding with obstacles. First, we define a diagonal judgment function $Diagonal()$ as (10) and two midpoint functions $C()$ and $D()$ as (11).

$$Diagonal(A, B) = |x_A - x_B| \wedge |y_A - y_B| \tag{10}$$

$$C(A, B) = (x_A, y_B) \quad , \quad D(A, B) = (x_B, y_A) \tag{11}$$

Next, we define the path insertion function. For a given path $P = (p_1, p_2, \ldots, p_n)$, inserting a point q at position i results in the following updated path, as expressed in (12).

$$Insert(P, i, q) = (p_1, \ldots, p_i, q, p_{i+1}, \ldots, p_n) \tag{12}$$

Similarly, we define the path deletion function. Given a path $P = (p_1, p_2, \ldots, p_n)$ and a set of points R to be removed, the updated path after deletion is given by (13).

$$Remove(P, R) = \{p \in P \mid p \notin R \vee p \in \{p_1, p_n\}\} \tag{13}$$

Finally, the corner avoidance optimization function $Optimize(P)$ is formulated as (14)–(17).

$$Optimize(P) = Remove(InsertPoints(P), R) \tag{14}$$

$$InsertPoints(P) = \bigcup_{i=1}^{n-1} Insert(P, i, q_i) \tag{15}$$

$$q_i = \begin{cases} C(p_i, p_{i+1}) & if\ C(p_i, p_{i+1}) \notin O, \\ D(p_i, p_{i+1}) & if\ D(p_i, p_{i+1}) \notin O, \\ \varnothing & otherwise. \end{cases} \tag{16}$$

$$R = \{p_i, p_{i+1} \mid Diagonal(p_i, p_{i+1}) \wedge (C(p_i, p_{i+1}) \in O \vee D(p_i, p_{i+1}) \in O)\} \tag{17}$$

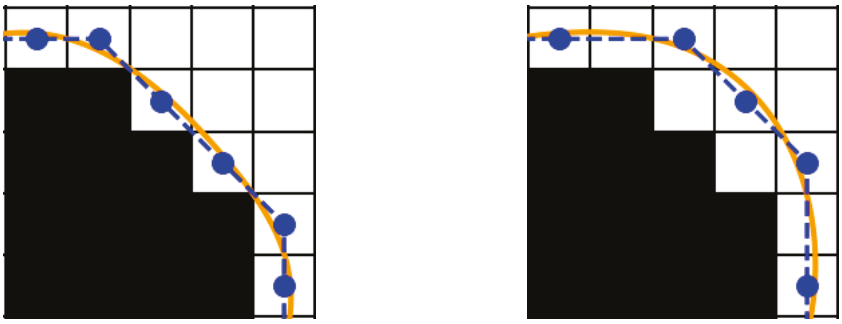
In (16) and (17), O represents the set of obstacles. The pseudo-code for corner avoidance is presented in Algorithm 2.

Algorithm 2: Corner Avoidance Strategy

Input: Grid map M, original path P
Output: Optimized path $P\prime$
BEGIN:
  Initialize $P' \leftarrow P.copy()$, $R \leftarrow \varnothing$
  $i \leftarrow 0$
  while $i < len(P\prime) - 1$ **do**
    $A \leftarrow P\prime[i]$, $B \leftarrow P\prime[i+1]$
    $x1, y1 \leftarrow A$; $x2, y2 \leftarrow B$
    if $|x1 - x2| == 1$ **and** $|y1 - y2| == 1$ **then** // Diagonal move
      $C \leftarrow (x1, y2)$, $D \leftarrow (x2, y1)$
      $has\_obstacle \leftarrow False$
      if $C \in M.obstacles \vee D \in M.obstacles$ **then** $has\_obstacle \leftarrow True$
      if $has\_obstacle$ **then**
      $R \leftarrow R \cup \{A, B\}$ // Mark nodes for removal
      if $C \notin M.obstacles$ **then**
        $P\prime.insert(i+1, C)$ // Insert intermediate point C
        $i \leftarrow i+1$ // Skip the new point
      else if $D \notin M.obstacles$ **then**
        $P'.insert(i+1, D)$ // Insert intermediate point D
        $i \leftarrow i+1$
      end if
      continue // Re-check current index
    end if
    end if
    $i \leftarrow i+1$
  end while
  // Remove marked nodes
  for p **in** P' **do**
    if $p \in R$ **and** $p \notin \{P[0], P[-1]\}$ **then**
      remove p from $P\prime$
    end if
  end for
  return P'
END

By applying $Optimize(P)$, the adjustment of control points enables obstacle avoidance during turning processes. This optimization ensures that the generated path smoothly navigates around obstacles rather than intersecting them. The effectiveness of this method is illustrated in Figure 3.

Path obstacle avoidance refers to the process of fine-tuning the positions of all control points based on nearby obstacles to ensure the entire path avoids obstacles effectively.



(a) (b)

Figure 3. The effect of corner avoidance optimization. (**a**) is the smoothed B-spline path; (**b**) is the improved smoothed B-spline path.

To achieve obstacle avoidance throughout the path, we adjust the positions of control points according to their proximity to obstacles. Therefore, we introduce the distance adjustment function $f(d_i)$ as (18), which modifies the position of a control point $P(x, y)$ when there are N nearby obstacles. Given an obstacle $O_i(x_i, y_i)$ located at a distance d_i, the adjusted position of the control point is denoted as D_i and expressed in (19) and (20).

$$f(d_i) = \begin{cases} \frac{1}{a+be^{d_i}} - c & d_i < d_0 \\ 0 & otherwise \end{cases} \tag{18}$$

$$\boldsymbol{u}_i = \frac{(x - x_i\ ,\ y - y_i)}{d_i} \tag{19}$$

$$\boldsymbol{D}_i = \frac{1}{N} \cdot f(d_i) \cdot \boldsymbol{u}_i \tag{20}$$

The pseudo-code for path obstacle avoidance is presented in Algorithm 3.

Algorithm 3: Path Obstacle Avoidance with Grid Map

Input: Optimized path P'; Grid map M; Distance threshold d_0; Parameters $a,\ b,\ c$
Output: Final control point path P''

BEGIN:
  Initialize P'' as empty list
  for p **in** P' **do**
    if $p == P'[0]$ **or** $p == P'[-1]$ **then**
      append p to p''
      continue
    end if
    Initialize $D_x \leftarrow 0,\ D_y \leftarrow 0,\ N \leftarrow 0$
  for each obstacle $O_i(x_i,\ y_i)$ **in** $M.obstacles$ **do**
    $d_i \leftarrow sqrt\left((x - x_i)^2 + (y - y_i)^2\right)$
    if $d_i < d_0$ **then**
      $f(d_i) \leftarrow 1/(a + b * e^x) - c$
      $D_x \leftarrow D_x + f(d_i) * (x - x_i)/d_i$
      $D_y \leftarrow D_y + f(d_i) * (y - y_i)/d_i$
      $N \leftarrow N + 1$
    end if
  end for
  if $N > 0$ **then**

Algorithm 3: *Cont.*

```
    P_adjusted x ← x + D_x / N
    P_adjusted y ← y + D_y / N
  else
    P_adjusted ← P
  end if
  append P_adjusted to P''
end for
return P''
END
```

By applying path obstacle avoidance to adjust control points, the B-spline smoothed path is optimized to ensure a safe distance from obstacles, enhancing the overall path safety.

4. Experimental Simulation

4.1. Path Planning Algorithm Simulation Experiment

4.1.1. Parameters for Path Planning Simulation

We use Python as the development environment. The simulation experiments are conducted in two types of map environments: high-density clustered maps and low-density random maps. Each environment is tested on maps of three different sizes: 20×20, 35×35, and 50×50.

For a given environment and map size, we first apply the SSA-A* to optimize the weight parameters and obtain the heuristic function $h(n)$. Then, we conduct path planning tests on 15 maps of the same environment and size. The paths generated by the SSA-A* are compared to those produced by Dijkstra's Algorithm, the Classical A* Algorithm, the JPS-A* Algorithm (an improved version of A* algorithm), and ACO.

The parameters used in the SSA optimization process are listed in Table 1.

Table 1. Parameters for path planning simulation experiments.

Parameter Name	Value or Range
ω_1	[0.5, 2]
ω_2	[0.5, 2]
ω_3	[0.1, 1]
Population Size (n)	10
Number of Iterations	100

4.1.2. Results and Analysis of Path Planning Simulation Experiment

Figure 4 shows the path planning simulation experimental results for a high-density clustered map on a 35×35 grid map. (a) corresponds to the path generated by the SSA-A*. (b) corresponds to the path generated by the Classical A* Algorithm. (c) corresponds to the path generated by Dijkstra's Algorithm. (d) corresponds to the path generated by the JPS-A* Algorithm. (e) corresponds to the path generated by ACO.

Figure 5 presents the path planning simulation experimental results for a low-density random map on a 35×35 grid map. (a) corresponds to the path generated by the SSA-A*. (b) corresponds to the path generated by the Classical A* Algorithm. (c) corresponds to the path generated by Dijkstra's Algorithm. (d) corresponds to the path generated by the JPS-A* Algorithm. (e) corresponds to the path generated by ACO.

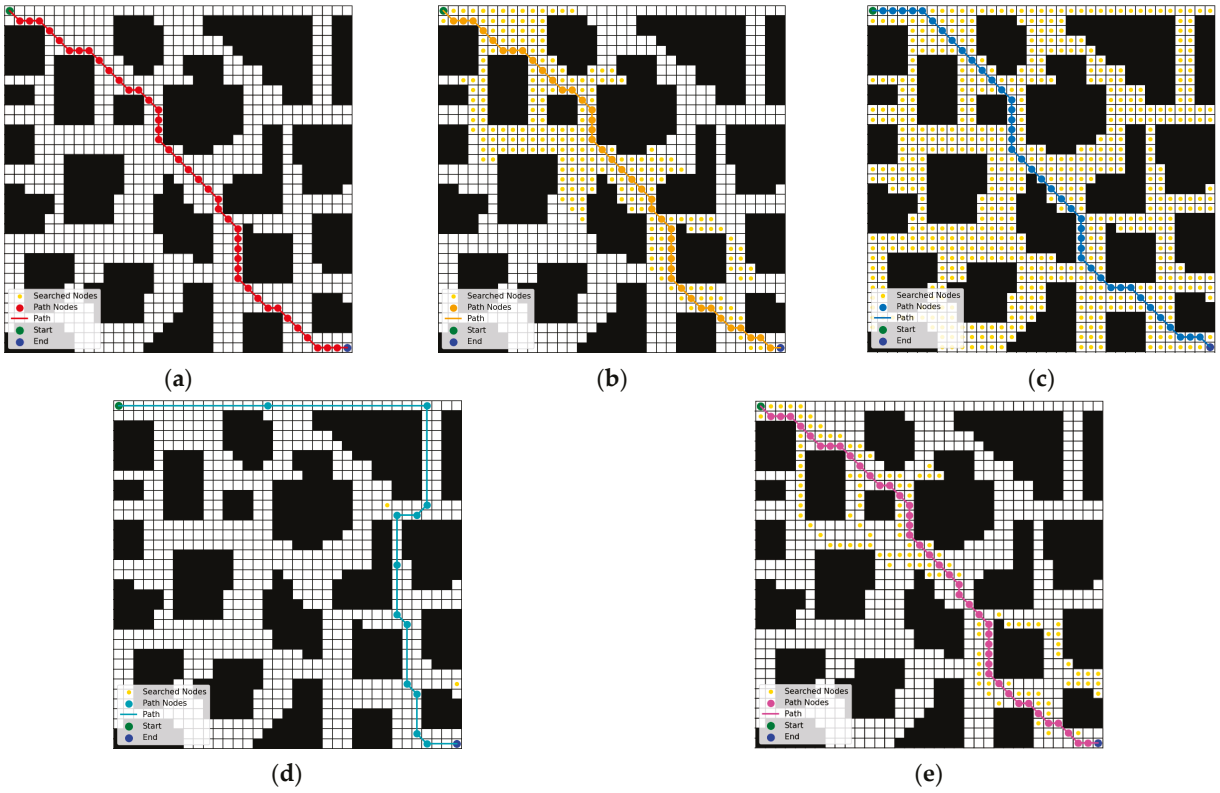


Figure 4. Simulation experimental results of five path planning algorithms on a high-density clustered map. (**a**) is the SSA-A*; (**b**) is the Classical A* Algorithm; (**c**) is Dijkstra's Algorithm; (**d**) is the JPS-A* Algorithm; (**e**) is ACO.

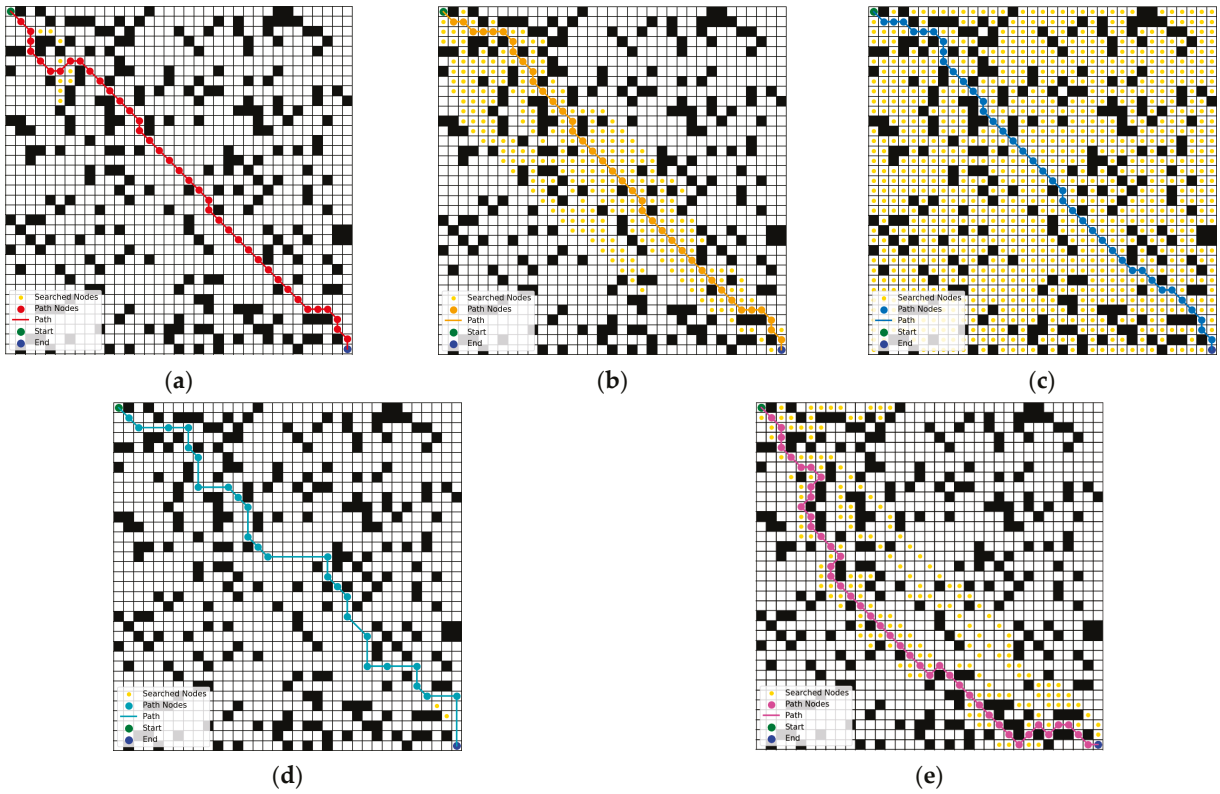


Figure 5. Simulation experimental results of five path planning algorithms on a low-density random map. (**a**) is the SSA-A*; (**b**) is the Classical A* Algorithm; (**c**) is Dijkstra's Algorithm; (**d**) is the JPS-A* Algorithm; (**e**) is ACO.

Table 2 shows the detailed data of path planning on a 35 × 35 grid map for both high-density clustered maps and low-density random maps. In this table, SN represents the number of search nodes.

Table 2. Path planning data for a 35 × 35 grid map.

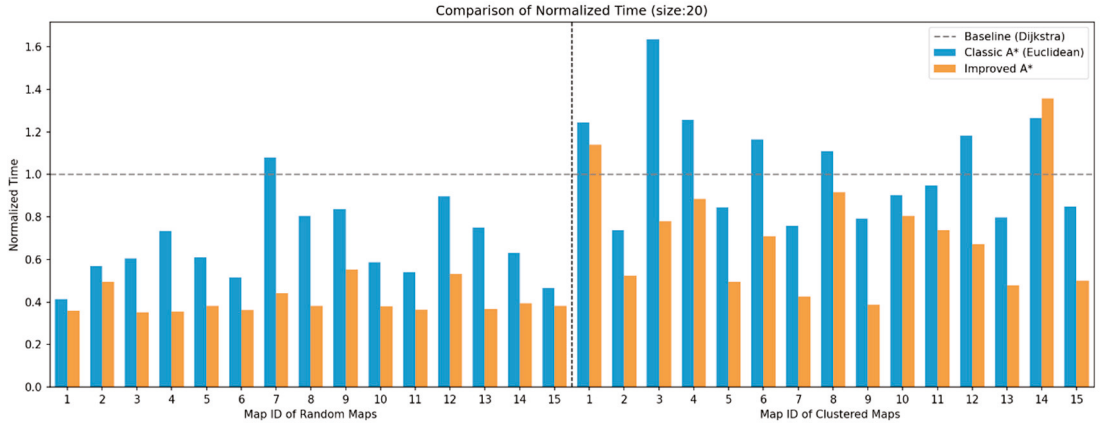
Map ID	Algorithm	Cluster Maps			Random Maps		
		SN	Length	Time (ms)	SN	Length	Time (ms)
1	SSA-A*	44	53.36	1.41	39	51.25	0.78
	A*	261	53.36	4.01	191	49.84	1.54
	Dijkstra	631	53.36	4.38	979	49.84	4.05
	JPS-A*	16	71.66	1.01	30	62.14	0.62
	ACO	235	54.18	24.15	234	50.43	26.55
2	SSA-A*	53	52.43	1.82	50	52.43	0.92
	A*	239	51.6	4.17	297	51.60	2.64
	Dijkstra	651	51.6	4.35	982	51.60	4.63
	JPS-A*	23	62.38	1.51	29	60.97	0.9
	ACO	248	51.6	31.31	190	54.43	23.33
3	SSA-A*	41	51.6	1.38	41	51.84	0.99
	A*	187	51.6	2.86	309	51.01	3.34
	Dijkstra	641	51.6	4.1	981	51.01	4.58
	JPS-A*	26	58.63	1.47	27	62.14	0.85
	ACO	191	52.43	28.24	206	53.25	24.77
4	SSA-A*	40	51.01	1.42	48	52.43	0.97
	A*	209	51.01	3.37	278	51.60	2.72
	Dijkstra	644	51.01	4.55	967	51.60	4.66
	JPS-A*	21	66.63	1.45	30	61.56	0.79
	ACO	233	51.6	31.85	235	55.50	25.22
5	SSA-A*	41	51.01	1.41	55	53.60	1.19
	A*	178	51.01	2.75	338	52.77	3.25
	Dijkstra	643	51.01	4.27	940	52.77	3.82
	JPS-A*	36	61.8	2.38	37	61.80	0.69
	ACO	205	52.43	31.69	137	58.08	12.26
6	SSA-A*	45	53.94	1.47	43	51.60	0.84
	A*	214	53.36	3.21	229	50.43	1.89
	Dijkstra	654	53.36	4.47	989	50.43	3.98
	JPS-A*	21	59.21	1.52	25	61.56	0.66
	ACO	295	54.77	33.05	170	50.43	29.83
7	SSA-A*	52	51.01	1.77	43	52.43	0.86
	A*	207	51.01	3.51	201	49.84	1.75
	Dijkstra	612	51.01	4.12	993	49.84	3.91
	JPS-A*	20	70.38	1.58	30	62.97	0.77
	ACO	251	51.84	24.81	168	51.25	29.35
8	SSA-A*	43	54.43	1.54	40	51.25	0.81
	A*	260	52.77	4.34	230	50.43	2.19
	Dijkstra	629	52.77	4.46	983	50.43	4
	JPS-A*	19	63.31	1.45	24	62.73	0.64
	ACO	242	56.43	34.47	230	61.74	24.88
9	SSA-A*	52	53.84	1.72	46	52.77	0.88
	A*	234	52.18	3.8	364	52.18	3.04
	Dijkstra	639	52.18	4.39	983	52.18	3.8
	JPS-A*	15	58.63	1.05	26	62.14	0.69
	ACO	292	60.08	36.16	202	57.84	28.28

Table 2. *Cont.*

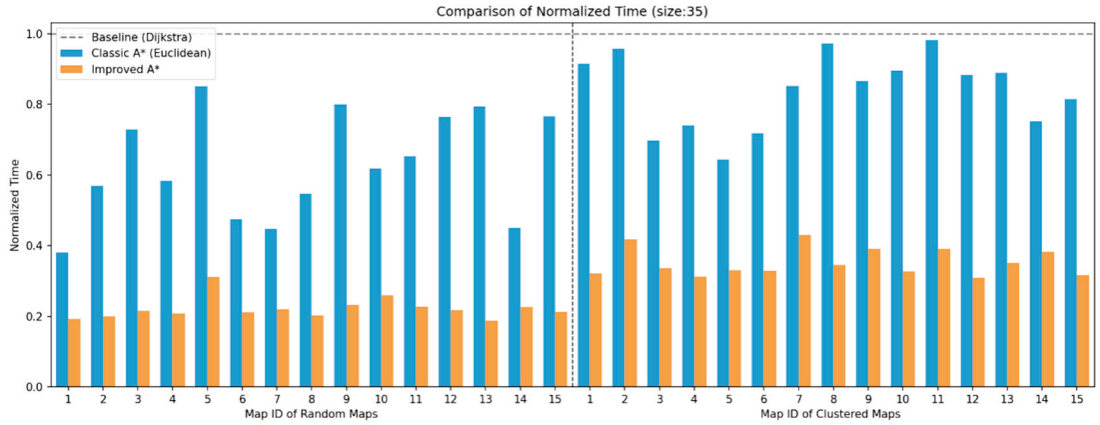
Map ID	Algorithm	Cluster Maps			Random Maps		
		SN	Length	Time (ms)	SN	Length	Time (ms)
10	SSA-A*	42	52.18	1.45	58	56.43	0.99
	A*	251	52.18	3.96	300	55.60	2.37
	Dijkstra	653	52.18	4.42	995	55.60	3.83
	JPS-A*	17	62.73	1.38	89	80.63	2.26
	ACO	220	52.18	31.52	273	60.67	29.63
11	SSA-A*	52	55.36	1.53	53	53.84	0.89
	A*	239	54.53	3.84	319	51.60	2.55
	Dijkstra	586	54.53	3.91	994	51.60	3.9
	JPS-A*	19	60.04	1.29	35	60.63	0.88
	ACO	177	56.18	35.74	233	53.84	25.41
12	SSA-A*	44	54.18	1.41	51	52.18	0.85
	A*	251	53.36	4.02	353	51.60	2.99
	Dijkstra	657	53.36	4.55	977	51.60	3.91
	JPS-A*	17	69.21	1.38	43	63.56	1.23
	ACO	192	55.01	36.54	242	54.18	24.47
13	SSA-A*	52	53.36	1.47	45	52.43	0.83
	A*	227	51.6	3.72	362	51.60	3.51
	Dijkstra	627	51.6	4.18	965	51.60	4.42
	JPS-A*	23	67.31	1.49	30	62.14	0.83
	ACO	169	53.84	33.62	239	54.67	26.32
14	SSA-A*	57	57.94	1.67	43	51.60	1.03
	A*	188	56.28	3.28	194	49.84	2.05
	Dijkstra	609	56.28	4.36	971	49.84	4.55
	JPS-A*	19	62.63	2.5	25	58.63	0.74
	ACO	171	58.77	36.19	187	52.08	26.83
15	SSA-A*	45	54.77	1.5	43	51.60	0.98
	A*	229	52.77	3.86	346	51.60	3.53
	Dijkstra	630	52.77	4.73	968	51.60	4.61
	JPS-A*	20	68.63	2	26	60.97	0.74
	ACO	189	57.6	32.46	251	52.43	25.72

Figure 6 shows the comparison of the running time between the SSA-A* and the Classical A* Algorithm. The running times of both algorithms are normalized with respect to the running time of Dijkstra's Algorithm (Dijkstra = 1). A lower value indicates a shorter running time.

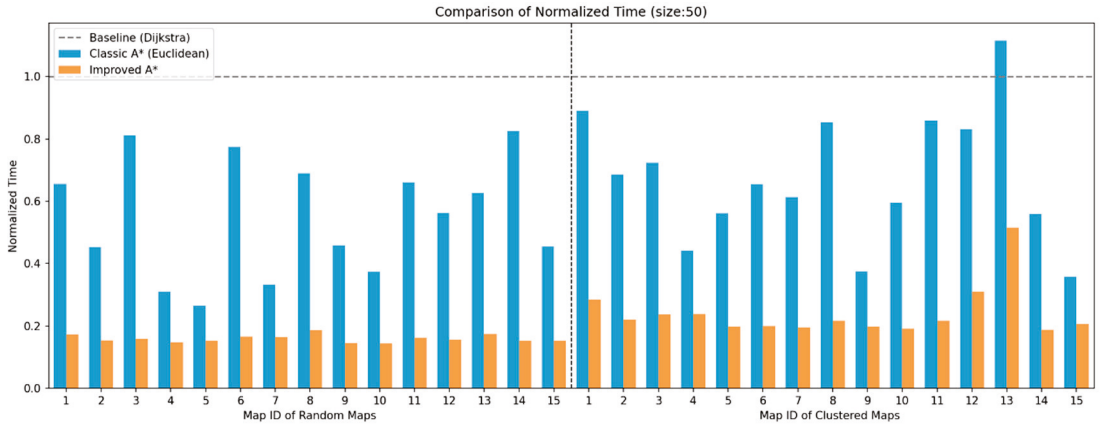
Figure 7 presents a comparison of the search node count and path length between the SSA-A* and Classical A* Algorithms. Search Efficiency (Left Axis): The number of search nodes for both the SSA-A* and Classical A* algorithms is normalized with respect to the number of search nodes in Dijkstra's Algorithm (Dijkstra = 1). A lower value indicates higher search efficiency. Path Quality (Right Axis): The SSA-A* path length increase (relative to the shortest path from Dijkstra) is shown as a percentage.



(**a**)



(**b**)



(**c**)

Figure 6. Comparison of running time between the SSA-A* and Classical A* Algorithms. (**a**) corresponds to the 20 × 20 grid map, (**b**) corresponds to the 35 × 35 grid map, and (**c**) corresponds to the 50 × 50 grid map.

Table 3 presents a comparison of the average performance of the SSA-A* against other algorithms across different map sizes.



Figure 7. Comparison of search node count and path length between the SSA-A* and Classical A* Algorithms. (**a**) corresponds to the 20 × 20 grid map, (**b**) corresponds to the 35 × 35 grid map, and (**c**) corresponds to the 50 × 50 grid map.

Based on the simulation experimental results, for the high-density clustered map, the SSA-A* reduces the running time by an average of 56.17% compared to Dijkstra's Algorithm, by 50.35% compared to the A* algorithm, and by 95.26% compared to ACO. The SSA-A* reduces the number of search nodes by an average of 91.41% compared to Dijkstra's Algorithm, by 75.97% compared to the A* algorithm, and by 73.91% compared to ACO.

Table 3. Comparison of the average performance of the SSA-A* and other algorithms.

Map Size		20		35		50	
Map Type		Cluster Map	Random Map	Cluster Map	Random Map	Cluster Map	Random Map
Average Time Reduction in SSA-A*	vs. A* (%)	30.79%	36.56%	57.45%	63.09%	62.82%	67.75%
	vs. Dijkstra (%)	27.89%	59.35%	64.69%	77.85%	75.94%	84.12%
	vs. ACO (%)	95.57%	96.92%	95.16%	96.16%	95.04%	97.61%
Average Searched Nodes Reduction in SSA-A*	vs. A* (%)	67.17%	72.52%	78.87%	83.08%	81.86%	86.10%
	vs. Dijkstra (%)	84.64%	91.01%	92.58%	95.19%	95.00%	96.73%
	vs. ACO (%)	64.32%	71.83%	78.03%	77.43%	79.37%	82.75%
Average Distance Increase in SSA-A* vs. A* (%)		1.62%	1.46%	1.48%	2.18%	0.58%	1.82%
Average Distance Reduction in SSA-A*	vs. JPS-A* (%)	14.85%	18.93%	22.24%	22.33%	15.00%	21.96%
	vs. ACO (%)	4.45%	4.72%	2.17%	3.76%	5.41%	5.00%

For the low-density random map, the SSA-A* reduces the running time by an average of 73.77% compared to Dijkstra's Algorithm, by 55.80% compared to the A* algorithm, and by 96.90% compared to ACO. The SSA-A* reduces the number of search nodes by an average of 94.31% compared to Dijkstra's Algorithm, by 80.57% compared to the A* Algorithm, and by 77.34% compared to ACO.

With the increase in map size, the advantages of the SSA-A* become more apparent. For example, on the 50 × 50 low-density random map, the SSA-A* reduces the running time by 67.75% compared to Dijkstra's Algorithm and by 84.12% compared to the A* algorithm. Additionally, the number of search nodes is reduced by 96.73% compared to Dijkstra's Algorithm and by 86.10% compared to the A* algorithm. In all cases, the JPS-A* algorithm shows an average increase in path length of 14% or more compared to the shortest path, while ACO shows an increase of 4.25%. In contrast, the SSA-A* increases the path length by only 0.5% to 2.2%, enhancing its suitability for practical applications.

4.2. Path Smoothing Algorithm Simulation Experiment

4.2.1. Parameters for the Path Smoothing Algorithm Simulation Experiment

Python was used as the development environment, and grid maps of sizes 20 × 20, 35 × 35, and 50 × 50 were selected. The obstacle density and distribution methods were set to high-density clustering and low-density random, respectively, to represent the entire environmental information. The maps were categorized into six groups according to the above standards, with each group consisting of 15 different maps for testing. In the maps, the green circle represents the start point, the blue circle represents the endpoint, the light blue line represents the initial path, the orange line represents the path after smoothing with the B-spline algorithm, and the red line represents the path after smoothing with the improved B-spline algorithm.

The parameters for the B-spline algorithm used in the simulation experiment are shown in Table 4.

Table 4. Parameter settings of B-spline interpolation.

Parameter Name	Parameter Value
Smoothing Factor (s)	0.1
Number of Interpolation Points	100

In B-spline interpolation, the smoothing factor is used to control the smoothness of the B-spline curve, while the number of interpolation points determines the accuracy and computational complexity of the B-spline curve. After repeated experiments, the smoothing factor was set to 0.1, and the number of interpolation points was set to 100.

4.2.2. Results and Analysis of the Path Smoothing Algorithm Simulation Experiment

Based on the aforementioned simulation environment and parameter settings, simulation experiments were conducted to calculate the smoothed paths for different algorithms. The results are presented for a high-density clustered map of size 20×20 (in the upper part of Figure 8) and a low-density random map (in the bottom part of Figure 8). As shown in (a) and (e), the original path contains multiple dangerous points near obstacles, with the distance between the path and obstacles being too small, which could lead to collisions. In (b) and (f), after B-spline interpolation smoothing, the path becomes noticeably smoother and more continuous, with no sharp corners; however, the path still remains too close to the obstacles. As shown in (c) and (g), after smoothing with the improved B-spline algorithm, the path not only maintains a safe distance from the obstacles but also remains continuous and smooth.

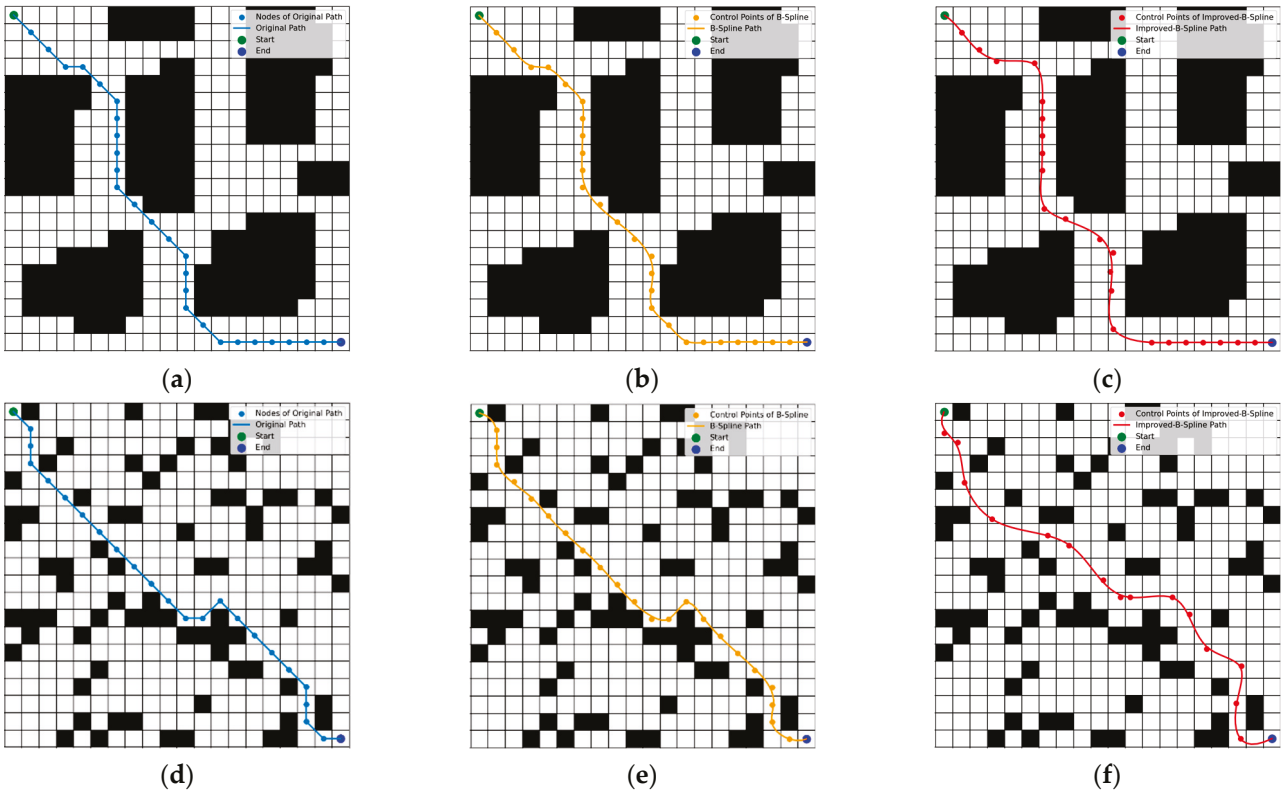


Figure 8. Smoothed paths of various algorithms. The first three figures are high-density maps with clustering. (**a**) represents the original path, (**b**) represents the path smoothed by B-spline, and (**c**) represents the path smoothed by the improved B-spline. The last three figures are low-density maps with random distribution. (**d**) represents the original path, (**e**) represents the path smoothed by B-spline, and (**f**) represents the path smoothed by the improved B-spline.

The performance of the original path, the path smoothed by the B-spline algorithm, and the path smoothed by the improved B-spline algorithm is evaluated below. The evaluation primarily focuses on two key metrics: the curvature of the path and the distance from the path to the nearest obstacle.

The logic for calculating the degree of meandering is as follows: As shown in Figure 9, adjacent points in the path are connected to form two vectors. The angle between these

vectors is calculated, and the rate of change in this angle is determined. This rate of change is defined as the degree of meandering. A smaller degree of meandering indicates a smoother path.

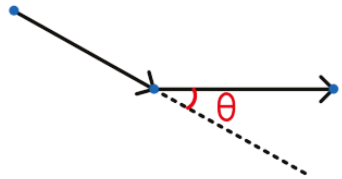


Figure 9. Method for calculating the degree of meandering.

The distance from the path to the obstacle is one of the important indicators for measuring the safety of the path. The greater the distance from the path to the obstacle, the higher the safety and reliability of the path. Here, the distance from each point on the path to the nearest obstacle is calculated to assess the safety of the path.

In Figure 10, (a) and (b) compare the curvature of the paths and the distance from the paths to the nearest obstacle for each algorithm on the clustered map displayed above. (c) and (d) compare the curvature of the paths and the distance of the paths to the nearest obstacle for each algorithm on the random map displayed above. As can be seen from the figures, after smoothing with the improved B-spline algorithm, the path maintains the largest distance from the nearest obstacle, with a moderate curvature, thereby meeting the requirements.

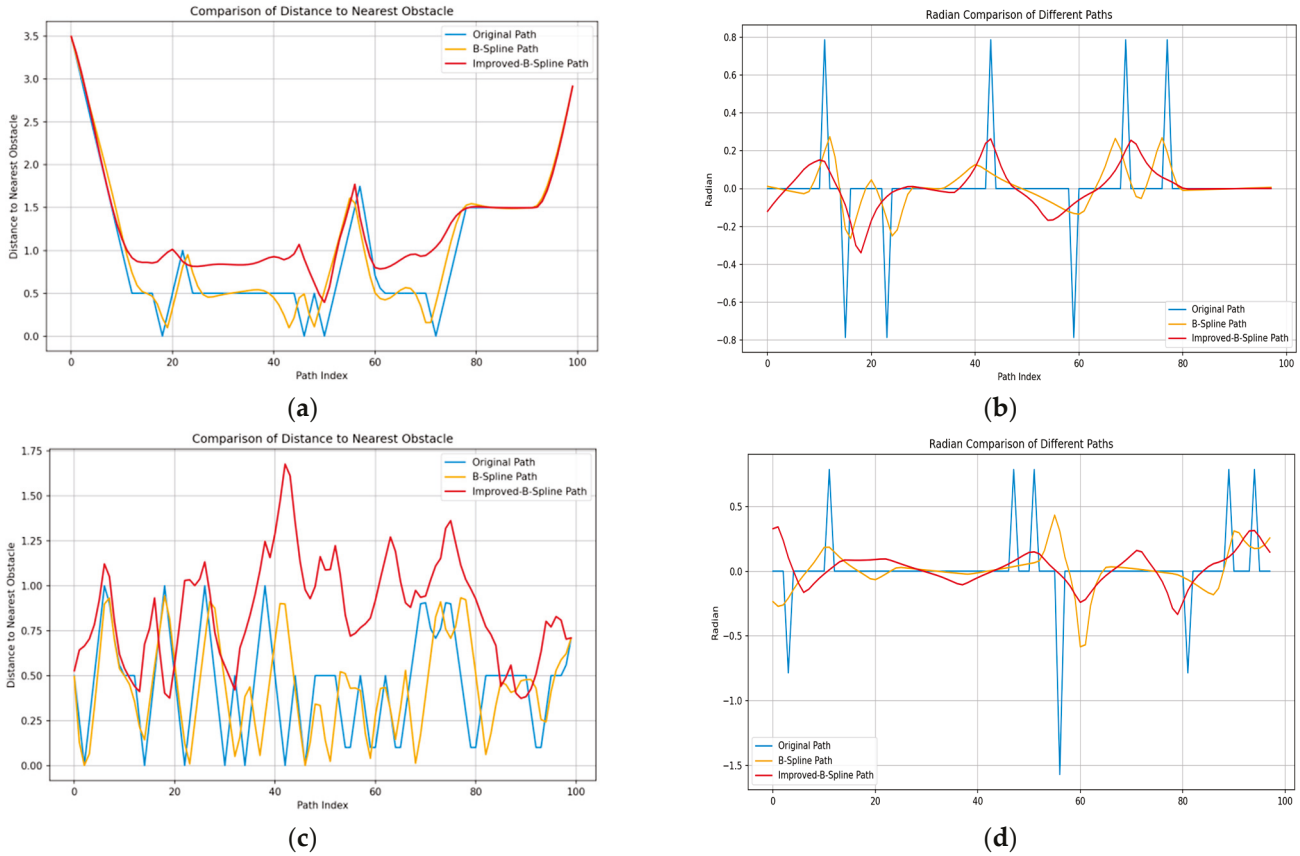


Figure 10. Performance comparison of each path. (**a**,**b**) represent the performance analysis of the high-density clustered map shown previously; (**c**,**d**) represent the performance analysis of the random low-density map shown previously.

Table 5 compares the minimum distance (MD) to the nearest obstacle, the average distance (AD) to the nearest obstacle, and the average curvature (AT) of the initial path (A), the path smoothed by the B-spline algorithm (B), and the path smoothed by the improved B-spline algorithm (C) for two types of maps with a size of 20 × 20.

Table 5. Performance comparison of each path.

Map ID	Algorithm	Cluster Maps			Random Maps		
		MD	AD	AT	MD	AD	AT
1	A	0.0000	0.6945	0.3521	0.0000	0.7765	0.4439
	B	0.0000	0.7063	0.1387	0.0060	0.7638	0.1452
	C	0.4564	1.0097	0.1270	0.2277	1.0051	0.1549
2	A	0.0000	0.6809	0.3073	0.0000	0.5451	0.3366
	B	0.0752	0.7149	0.0792	0.0213	0.5507	0.0732
	C	0.4781	1.0321	0.1014	0.1000	0.7996	0.1575
3	A	0.0000	1.0024	0.2115	0.0000	0.7057	0.3142
	B	0.0961	1.0423	0.0668	0.0000	0.6960	0.0746
	C	0.3939	1.2601	0.0730	0.0480	1.0206	0.1132
4	A	0.0000	0.9127	0.3600	0.1000	1.1887	0.1653
	B	0.0046	0.9452	0.1102	0.0000	1.2021	0.0252
	C	0.8040	1.2919	0.1340	0.2677	1.3819	0.0469
5	A	0.0000	0.7055	0.2618	0.0000	0.6499	0.3927
	B	0.0183	0.7269	0.0994	0.0000	0.6707	0.1102
	C	0.3346	0.9950	0.0964	0.4718	0.9050	0.0967
6	A	0.0000	0.7267	0.2827	0.0000	0.7176	0.3366
	B	0.0919	0.7426	0.0915	0.0000	0.7250	0.0857
	C	0.7542	1.0624	0.1014	0.0975	0.9804	0.1088
7	A	0.0000	0.6324	0.1870	0.0000	0.6256	0.2356
	B	0.0127	0.6415	0.0532	0.0000	0.6289	0.0569
	C	0.3643	0.9650	0.1250	0.1845	0.9347	0.1308
8	A	0.0000	0.7558	0.4254	0.0000	0.4808	0.4254
	B	0.0260	0.7715	0.1350	0.0084	0.4984	0.1336
	C	0.6290	1.1425	0.1471	0.0974	0.6789	0.1402
9	A	0.0000	0.6990	0.3142	0.0000	0.6392	0.2992
	B	0.0000	0.7113	0.1155	0.0000	0.6500	0.0748
	C	0.4775	1.0443	0.1069	0.2611	0.9248	0.1339
10	A	0.0000	0.8205	0.2856	0.0000	0.6837	0.2356
	B	0.0084	0.8199	0.0793	0.0000	0.6818	0.0546
	C	0.6038	1.1713	0.0950	0.3946	1.0015	0.1122
11	A	0.0000	0.7276	0.3200	0.0000	0.7396	0.2618
	B	0.0277	0.7350	0.1053	0.0227	0.7692	0.0760
	C	0.5741	1.0648	0.1206	0.3956	0.9740	0.1391
12	A	0.1000	0.7944	0.2618	0.0000	0.4960	0.3927
	B	0.0359	0.8430	0.0541	0.0221	0.4932	0.1104
	C	0.3501	1.0428	0.0859	0.3259	0.8577	0.1283
13	A	0.0000	0.7710	0.2291	0.0000	0.5484	0.3756
	B	0.0458	0.7907	0.0758	0.0000	0.5329	0.1150
	C	0.7377	1.0929	0.0833	0.0731	0.7502	0.1057

Table 5. *Cont.*

Map ID	Algorithm	Cluster Maps			Random Maps		
		MD	AD	AT	MD	AD	AT
14	A	0.0000	0.9868	0.1636	0.0000	0.9388	0.3366
	B	0.0702	0.9931	0.0487	0.0606	0.9384	0.1163
	C	0.4245	1.2563	0.0641	0.1057	1.1389	0.1701
15	A	0.0000	0.5984	0.2199	0.0000	0.5090	0.2618
	B	0.0365	0.6015	0.0754	0.0419	0.5247	0.0702
	C	0.3309	0.9086	0.0961	0.2562	0.8942	0.1368

The following conclusions can be drawn:

In the high-density clustered map, the average minimum distance from the smoothed path of the improved B-spline algorithm to the nearest obstacle increased by 0.5142 compared to the initial path and by 0.4776 compared to the B-spline algorithm's smoothed path. The average distance from the improved B-spline algorithm smoothed path to the nearest obstacle increased by 43.05% compared to the initial path and by 39.82% compared to the B-spline algorithm smoothed path. The average curvature of the improved algorithm smoothed path decreased by 61.87% compared to the initial path.

In the low-density random map, the average minimum distance from the improved B-spline algorithm's smoothed path to the nearest obstacle increased by 0.2137 compared to the initial path and by 0.2082 compared to the B-spline algorithm's smoothed path. The average distance from the improved B-spline algorithm's smoothed path to the nearest obstacle increased by 42.20% compared to the initial path and by 41.04% compared to the B-spline algorithm's smoothed path. The average curvature of the improved algorithm's smoothed path decreased by 59.96% compared to the initial path.

Figure 11 shows the performance comparison of all maps in the test set, categorized by size. It presents line graphs of the minimum distance to the nearest obstacle, the average distance to the nearest obstacle, and the average curvature for two types of maps with a size of 20×20. The same classification and visualization are applied to the other two map sizes. From the figure, it is clearly observed that the distance from the path smoothed by the improved B-spline algorithm to the nearest obstacle is always the largest, the average distance to the nearest obstacle is also the largest, and the average curvature is moderate, significantly reduced compared to the original path.

4.3. Benchmark Tests

The tests were conducted using the city/street maps dataset from the MovingAI 2D Pathfinding Benchmarks, running on hardware with a Ryzen 7 6800H/RTX 3060 Laptop GPU configuration, with the algorithm implemented in Python 3.12.

Figure 12 presents the results of benchmark tests on a 256×256 map. (a) corresponds to the path generated by the SSA-A*. (b) corresponds to the path generated by the Classical A* Algorithm. (c) corresponds to the path generated by Dijkstra's Algorithm. (d) corresponds to the path generated by the JPS-A* Algorithm. (e) corresponds to the path generated by ACO.

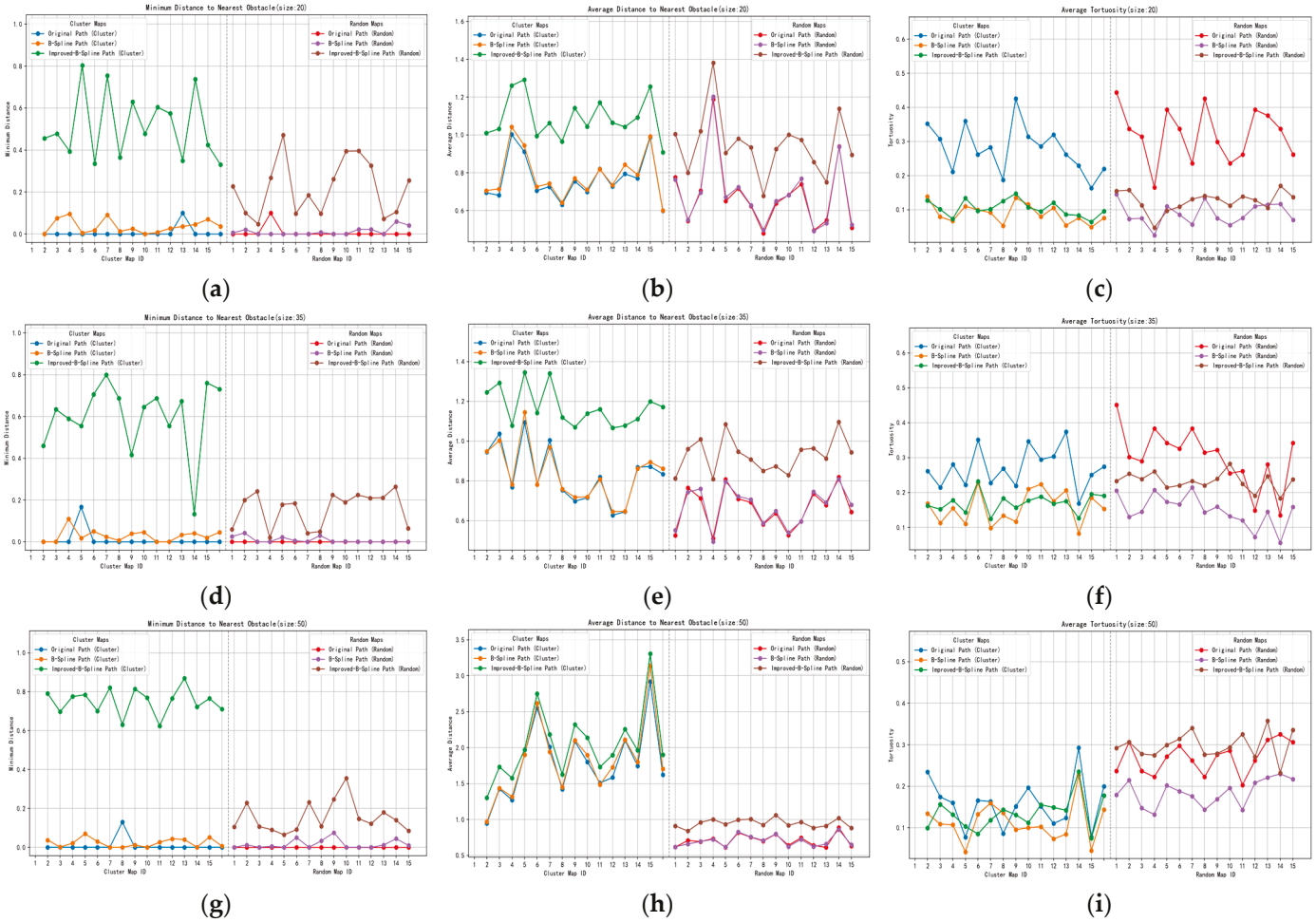


Figure 11. Performance comparison of the test set. (**a**–**c**) present line graphs of the minimum distance to the nearest obstacle, the average distance to the nearest obstacle, and the average curvature for two types of maps with a size of 20 × 20, respectively; (**d**–**f**) present line graphs of the minimum distance to the nearest obstacle, the average distance to the nearest obstacle, and the average curvature for two types of maps with a size of 35 × 35, respectively; (**g**–**i**) present line graphs of the minimum distance to the nearest obstacle, the average distance to the nearest obstacle, and the average curvature for two types of maps with a size of 50 × 50, respectively.

Table 6 presents the performance comparison results of various path planning algorithms in benchmark tests. The test results demonstrate that the SSA-A* reduces pathfinding time by 93.69% compared to the A* algorithm and by 93.30% compared to Dijkstra's Algorithm. The paths planned by the SSA-A*, JPS-A* algorithm, and ACO exhibit increases of 4.08%, 30.75%, and 32.65%, respectively, relative to the shortest path, while their average runtime accounts for 6.31%, 352.38%, and 210.45% of the A* algorithm's runtime. This discrepancy arises because the JPS -A* algorithm incurs redundant computations due to its jump strategy in real-world scenarios with irregular obstacles, leading to inefficiency and longer planned paths. Meanwhile, the ACO tends to select suboptimal branches in early iterations due to randomness, resulting in circuitous paths. In contrast, the SSA-A* demonstrates superior performance in both path length and planning time.

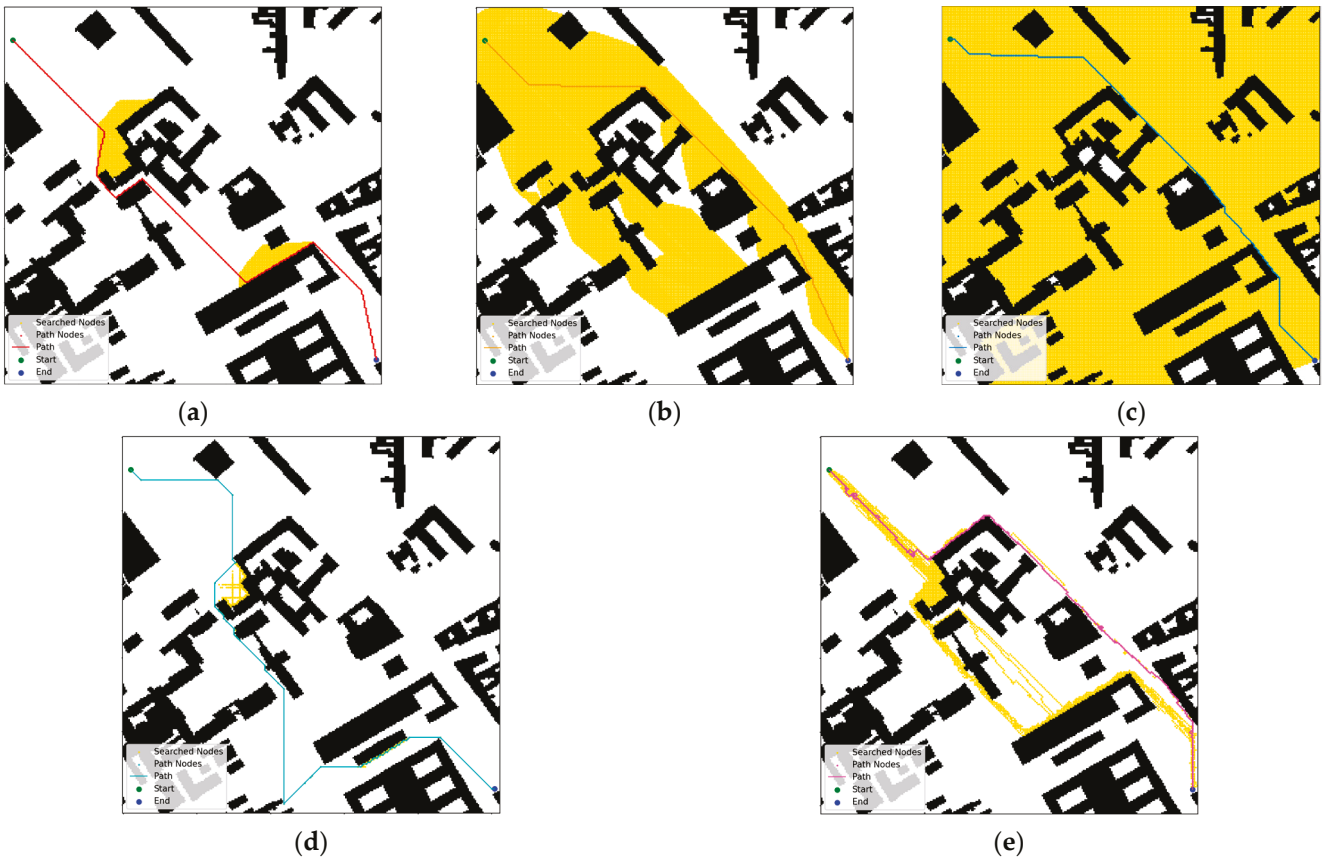


Figure 12. Results of benchmark tests of five path planning algorithms on a 256 × 256 map. (**a**) is the SSA-A*; (**b**) is the Classical A* Algorithm; (**c**) is Dijkstra's Algorithm; (**d**) is the JPS-A* Algorithm; (**e**) is ACO.

Table 6. Performance comparison of path planning algorithms in benchmark tests.

Algorithm	Comparison to A* (100%)		
	Searched-Nodes (%)	Length (%)	Time (%)
SSA-A*	4.39	104.83	6.31
Dijkstra	320.17	100.00	94.08
A*-JPS	0.70	130.75	325.38
ACO	12.04	132.65	210.45

Figure 13 presents the benchmark test results of five path smoothing algorithms on a 256 × 256 map. (a) and (d) show the global view and local details of the original path; (b) and (e) display the globally smoothed path and its local section after B-spline interpolation; (c) and (f) demonstrate the panoramic and zoomed-in views of the path processed by the improved B-spline interpolation smoothing algorithm.

For the path smoothing algorithms, the test results indicate that the Improved B-spline Path increases the minimum distance to the nearest obstacle by 0.78 compared to the Original Path and by 0.77 compared to the B-spline Path. Additionally, the average distance to the nearest obstacle for the Improved B-spline Path increases by 17.40% compared to the Original Path and by 14.93% compared to the B-Spline Path. Furthermore, the average curvature of the Improved B-spline Path decreases by 53.86% compared to the Original Path and by 28.54% compared to the B-spline Path.

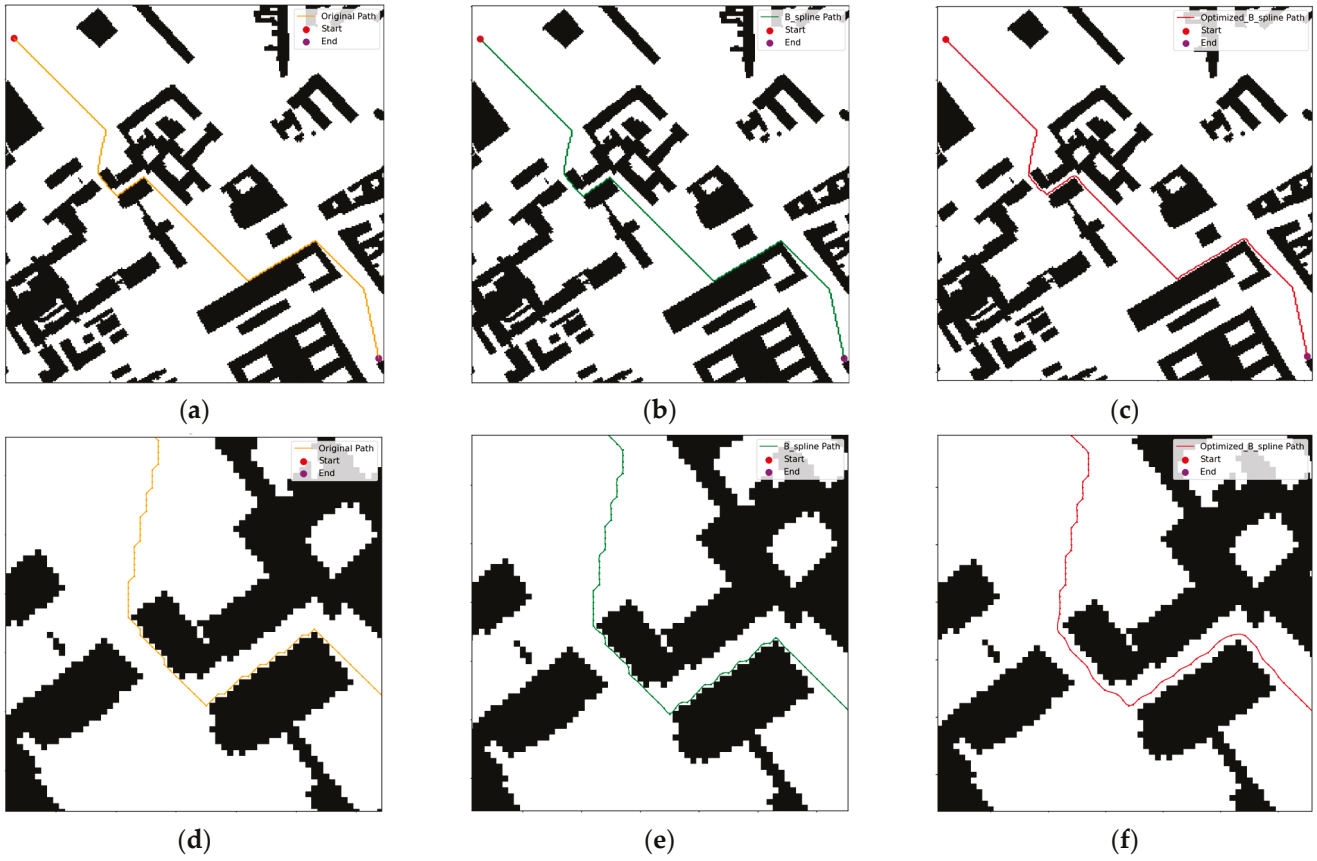


Figure 13. Results of benchmark tests of five path smoothing algorithms on a 256 × 256 map. (**a**) represents the original path, (**d**) is a detailed view of a portion of (**a**); (**b**) represents the path smoothed by B-spline, (**e**) is a detailed view of a portion of (**b**); (**c**) represents the path smoothed by the improved B-spline, and (**f**) is a detailed view of a portion of (**c**).

5. Conclusions

To address the limitations of the traditional A* algorithm, such as low performance and unsmooth paths in path planning, this paper proposes a scheme that improves the A* algorithm by integrating the SSA for path planning, followed by path smoothing using the improved B-spline interpolation method. Through comparative experiments in various scenarios, including high-density clustered environments and low-density random environments, and multiple grid map scales (ranging from 20 × 20 to 50 × 50), the optimized algorithm demonstrates significant advantages in key indicators such as search efficiency, smoothness, and safety, offering a more efficient and reliable path planning approach for practical application scenarios.

However, the SSA-A* still has certain limitations. While the SSA greatly enhances the optimization of the heuristic function, its performance can vary based on the complexity and specific characteristics of the environment. In environments with dynamic obstacles that change frequently, the algorithm may require more frequent re-optimization to adjust to new conditions. Additionally, the current implementation assumes a static environment, and extending it to handle real-time dynamic obstacle environments or 3D path planning requires further adjustments.

Future research could focus on addressing these limitations. One potential direction is the development of adaptive mechanisms that dynamically adjust the SSA parameters based on real-time environmental changes, thereby improving the algorithm's robustness in dynamic and real-time scenarios. Another approach could involve exploring hybrid algorithms that integrate the SSA-A* with other advanced techniques, such as reinforce-

ment learning or machine learning models, to further enhance path planning performance. Additionally, expanding the algorithm's scalability to handle more complex 3D path planning scenarios could be another key area for future development. In conclusion, while the SSA-A* offers a promising solution for path planning, continuous research is essential to overcome its current limitations and expand its applicability to a broader range of real-world environments.

Author Contributions: Conceptualization, H.Z., T.S. and L.Z.; Methodology, H.Z. and T.S.; Validation, H.Z., T.S. and Y.W.; Investigation, T.S. and Y.W.; Data curation, H.Z., T.S. and Y.W.; Writing—original draft, H.Z., T.S. and Y.W.; Writing—review & editing, H.Z., T.S. and Y.W.; Visualization, H.Z., T.S. and Y.W.; Supervision, L.Z.; Project administration, L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hassan, M.U.; Ullah, M.; Iqbal, J. Towards Autonomy in Agriculture: Design and Prototyping of a Robotic Vehicle with Seed Selector. In Proceedings of the 2016 2nd International Conference on Robotics and Artificial Intelligence (ICRAI), Rawalpindi, Pakistan, 1–2 November 2016; pp. 37–44. [CrossRef]
2. Liu, L.; Wang, X.; Yang, X.; Liu, H.; Li, J.; Wang, P. Path Planning Techniques for Mobile Robots: Review and Prospect. *Expert Syst. Appl.* **2023**, *227*, 120254. [CrossRef]
3. Bento, L.M.; Boccardo, D.R.; Machado, R.C.; Miyazawa, F.K.; de Sá, V.G.P.; Szwarcfiter, J.L. Dijkstra graphs. *Discret. Appl. Math.* **2017**, *261*, 52–62. [CrossRef]
4. Xiang, D.; Lin, H.; Ouyang, J.; Huang, D. Combined Improved A* and Greedy Algorithm for Path Planning of Multi-Objective Mobile Robot. *Sci. Rep.* **2022**, *12*, 13273. [CrossRef]
5. Zhang, T.; Xu, S.; Gao, Y.; Wang, P.; Schonfeld, P.; Zou, Y.; He, Q. 3D Constrained Hybrid A*: Improved Vehicle Path Planning Algorithm for Cost-Effective Road Alignment Design. *Autom. Constr.* **2024**, *166*, 105645. [CrossRef]
6. Huang, T.; Fan, K.; Sun, W. Density Gradient-RRT: An Improved Rapidly Exploring Random Tree Algorithm for UAV Path Planning. *Expert Syst. Appl.* **2024**, *252*, 124121. [CrossRef]
7. Huang, Z.; Chen, H.; Pohovey, J.; Driggs-Campbell, K. Neural Informed RRT*: Learning-based Path Planning with Point Cloud State Representations under Admissible Ellipsoidal Constraints. *arXiv* **2023**, arXiv:2309.14595. [CrossRef]
8. Worring, A.; Mayer, B.E.; Hamacher, K. Genetic Algorithm Niching by (Quasi-)infinite Memory. In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '21), Lille, France, 10–14 July 2021; pp. 296–304. [CrossRef]
9. García, M.; López, N.; Rodríguez, I. A Full Process Algebraic Representation of Ant Colony Optimization. *Inf. Sci.* **2024**, *658*, 120025. [CrossRef]
10. Wu, C.; Zhu, Z.; Tian, Z.; Zou, C. Generational Exclusion Particle Swarm Optimization with Central Aggregation Mechanism. In *Advances in Swarm Intelligence*; Tan, Y., Shi, Y., Niu, B., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2022; Volume 13344, p. 15. [CrossRef]
11. Li, X.; Wang, L.; An, Y.; Huang, Q.L.; Cui, Y.H.; Hu, H.S. Dynamic Path Planning of Mobile Robots Using Adaptive Dynamic Programming. *Expert Syst. Appl.* **2024**, *235*, 121112. [CrossRef]
12. Xu, Y.; Li, Q.; Xu, X.; Yang, J.; Chen, Y. Research Progress of Nature-Inspired Metaheuristic Algorithms in Mobile Robot Path Planning. *Electronics* **2023**, *12*, 3263. [CrossRef]
13. Johansson, A.; Markdahl, J. Swarm Bug Algorithms for Path Generation in Unknown Environments. In Proceedings of the 2023 62nd IEEE Conference on Decision and Control (CDC), Singapore, 13–15 December 2023; pp. 6958–6965.
14. Wang, M.; Xu, J.; Zhang, J.; Cui, Y. An Autonomous Navigation Method for Orchard Rows Based on a Combination of an Improved A* Algorithm and SVR. *Precis. Agric.* **2024**, *25*, 429–453. [CrossRef]
15. Wu, L.; Huang, X.; Cui, J.; Liu, C.; Xiao, W. Modified adaptive ant colony optimization algorithm and its application for solving path planning of mobile robot. *Expert Syst. Appl.* **2023**, *215*, 119410. [CrossRef]

16. Zheng, L.; Duan, H.; Huo, M.; Wu, H. Enhancing the Heuristic Function of Improved A Algorithm for UAV Robotic Arm Path Planning Using Dynamic Pigeon-Inspired Optimization. In *Advances in Guidance, Navigation and Control: ICGNC 2024*; Yan, L., Duan, H., Deng, Y., Eds.; Lecture Notes in Electrical Engineering; Springer: Singapore, 2025; Volume 1343, p. 14. [CrossRef]
17. Zhang, Q.; Liu, X.; Peng, L.; Zhu, F. Path Planning for Mobile Robots Based on JPS and Improved A* Algorithm. *J. Front. Comput. Sci. Technol.* **2021**, *15*, 2233–2240.
18. Huang, J.; Chen, C.; Shen, J.; Liu, G.; Xu, F. A Self-adaptive Neighborhood Search A-star Algorithm for Mobile Robots Global Path Planning. *Comput. Electr. Eng.* **2025**, *123*, 110018. [CrossRef]
19. Fan, Q.; Chen, Z.; Zhang, W.; Fang, X. ESSAWOA: Enhanced Whale Optimization Algorithm Integrated with Salp Swarm Algorithm for Global Optimization. *Eng. Comput.* **2022**, *38* (Suppl. S1), 797–814. [CrossRef]
20. Benmamoun, Z.; Khlie, K.; Bektemyssova, G.; Dehghani, M.; Gherabi, Y. Bobcat Optimization Algorithm: An Effective Bio-Inspired Metaheuristic Algorithm for Solving Supply Chain Optimization Problems. *Sci. Rep.* **2024**, *14*, 20099. [CrossRef]
21. Benmamoun, Z.; Khlie, K.; Dehghani, M.; Gherabi, Y. WOA: Wombat Optimization Algorithm for Solving Supply Chain Optimization Problems. *Mathematics* **2024**, *12*, 1059. [CrossRef]
22. Sang, H.; You, Y.; Sun, X.; Zhou, Y.; Liu, F. The Hybrid Path Planning Algorithm Based on Improved A* and Artificial Potential Field for Unmanned Surface Vehicle Formations. *Ocean Eng.* **2021**, *223*, 108709. [CrossRef]
23. Song, B.; Wang, Z.; Zou, L. An Improved PSO Algorithm for Smooth Path Planning of Mobile Robots Using Continuous High-degree Bezier Curve. *Appl. Soft Comput.* **2021**, *100*, 106960. [CrossRef]
24. Sun, W.; Sun, P.; Ding, W.; Zhao, J.; Li, Y. Gradient-based Autonomous Obstacle Avoidance Trajectory Planning for B-spline UAVs. *Sci. Rep.* **2024**, *14*, 14458. [CrossRef]
25. Hart, P.E.; Nilsson, N.J.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
26. Abualigah, L.; Shehab, M.; Alshinwan, M.; Alabool, H. Salp Swarm Algorithm: A Comprehensive Survey. *Neural Comput. Appl.* **2020**, *32*, 11195–11215. [CrossRef]
27. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
28. Nautiyal, B.; Prakash, R.; Vimal, V.; Liang, G.; Chen, H. Improved Salp Swarm Algorithm with Mutation Schemes for Solving Global Optimization and Engineering Problems. *Eng. Comput.* **2022**, *38*, 3927–3949. [CrossRef]
29. Guo, X.; Zhao, D.; Fan, T.; Long, F.; Fang, C.; Long, Y. Autonomous Underwater Vehicle Path Planning Based on Improved Salp Swarm Algorithm. *J. Mar. Sci. Eng.* **2024**, *12*, 1446. [CrossRef]
30. AlShabi, M.A.; Ballous, K.; Nassif, A.B.; Bettayeb, M.; Obaideen, K.; Gadsden, S.A. Path Planning for a UGV using Salp Swarm Algorithm. In Proceedings of the Autonomous Systems: Sensors, Processing, and Security for Ground, Air, Sea, and Space Vehicles and Infrastructure, National Harbor, MD, USA, 21–26 April 2024; Volume 13052.
31. Tang, S. Control Point Reduction for Lofted B-spline Surface Interpolation Based on Serial Closed Contours. *Comput. Appl. Math.* **2023**, *42*, 86. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

Advanced Cooperative Formation Control in Variable-Sweep Wing UAVs via the MADDPG–VSC Algorithm

Zhengyang Cao [1,2] and Gang Chen [1,*]

[1] State Key Laboratory of Strength and Vibration for Mechanic Structures, School of Aerospace Engineering, Xi'an Jiaotong University, Xi'an 710049, China; caozhengyang@stu.xjtu.edu.cn
[2] Xi'an ASN UAV Technology Co., Ltd., Xi'an 710065, China
* Correspondence: aachengang@xjtu.edu.cn

Abstract: UAV technology is advancing rapidly, and variable-sweep wing UAVs are increasingly valuable because they can adapt to different flight conditions. However, conventional control methods often struggle with managing continuous action spaces and responding to dynamic environments, making them inadequate for complex multi-UAV cooperative formation control tasks. To address these challenges, this study presents an innovative framework that integrates dynamic modeling with morphing control, optimized by the multi-agent deep deterministic policy gradient for two-sweep control (MADDPG–VSC) algorithm. This approach enables real-time sweep angle adjustments based on current flight states, significantly enhancing aerodynamic efficiency and overall UAV performance. The precise motion state model for wing morphing developed in this study underpins the MADDPG–VSC algorithm's implementation. The algorithm not only optimizes multi-UAV formation control efficiency but also improves obstacle avoidance, attitude stability, and decision-making speed. Extensive simulations and real-world experiments consistently demonstrate that the proposed algorithm outperforms contemporary methods in multiple aspects, underscoring its practical applicability in complex aerial systems. This study advances control technologies for morphing-wing UAV formation and offers new insights into multi-agent cooperative control, with substantial potential for real-world applications.

Keywords: multi-UAV; variable-sweep wing; multi-agent deep reinforcement learning; formation control; MADDPG–VSC

1. Introduction

As unmanned aerial vehicle (UAV) technology continues to advance rapidly, the value of variable-sweep wing UAVs has become increasingly evident, particularly due to their ability to adapt to a wide range of flight states, which significantly enhances overall performance and efficiency [1]. Traditional trajectory planning algorithms, such as A*, rapidly-exploring random trees (RRT), and probabilistic roadmaps (PRM), have been widely employed in UAV systems due to their effectiveness in structured environments with known dynamics and constraints. However, these classical methods often face challenges when applied to high-dimensional spaces, dynamic obstacles, and real-time applications, where their computational complexity and adaptability are limited [2–4]. This growing recognition highlights the critical importance of advancing research in variable-sweep wing technology through more adaptive control mechanisms, such as deep reinforcement learning (DRL). However, despite their potential, current research has not fully optimized these systems, particularly in the implementation of advanced control mechanisms such as the proximal policy optimization (PPO) algorithm [5]. With the expanding application of multi-UAV systems across commercial, environmental, and defense sectors, where coordinated operation is crucial, optimizing variable-sweep wing UAVs has become imperative [6]. This study seeks to integrate variable-sweep wing technology with advanced control algorithms,

aiming to enhance multi-UAV system capabilities by providing more flexible and efficient solutions, especially for formation and cooperative control tasks.

In the area of formation control, variable-sweep wing UAVs face significant challenges, particularly in achieving precise control and maintaining formation. Early studies, such as those by Valasek et al. [7] and Lampton et al. [8,9], employed Q-learning to optimize formation control for morphing UAVs based on aerodynamic characteristics. While these methods showed some effectiveness, their application was limited by the discrete nature of state and action spaces. Valasek et al. [10] applied deep reinforcement learning to simplified morphing aircraft models, achieving some progress in formation control; however, the simplifications in the models constrained the precise definition and execution of control actions. Xu et al. [11] advanced the field by using the deep deterministic policy gradient (DDPG) algorithm to control multiple biomimetic morphing UAVs, but this approach still lacks flight validation to support the optimization of formation control. Yan et al. [12] explored the adaptive control of UAV wing configurations using Q-learning, but these studies have yet to establish a direct link between formation control strategies and actual improvements in flight performance.

In the domain of cooperative control, existing methods such as Q-learning and deep Q-network (DQN) algorithms [13,14] have been widely applied in UAV systems, but they exhibit significant limitations when dealing with continuous action spaces and dynamic environments. Specifically, the discrete nature of state and action spaces in Q-learning restricts its adaptability, making it incapable of addressing the complexities of dynamic cooperative control tasks. While DQN is more effective in handling high-dimensional control tasks, it still faces challenges in complex dynamic environments, particularly in regards to multi-UAV cooperative formation control [15,16]. These methods often struggle to maintain stability and efficiency in scenarios requiring real-time responsiveness and robust decision making [17]. Therefore, there is an urgent need for advanced algorithms capable of overcoming these challenges, particularly in the context of variable-sweep wing UAVs, which demand precise and adaptive control mechanisms.

To tackle these challenges, this study presents the following key innovations:

(1) Optimization of the multi-rigid-body dynamic model for an adaptive morphing wing.

A novel optimization model for the multi-rigid-body dynamics of the variable-sweep wing UAV is introduced. Through precise model adjustments and dynamic optimization, the method significantly enhances the UAV's sweep angle adjustment and dynamic response capabilities across various flight states, providing robust theoretical support for attitude control during complex flight missions. Unlike simplified linearized models commonly used in UAV control, which overlook mass distribution and inertia variations, the proposed model demonstrates enhanced real-time adaptability, resulting in improved control precision across diverse flight conditions.

(2) Development of a cooperative control algorithm for variable-sweep wing UAVs.

An enhanced multi-agent deep deterministic policy gradient for the variable-sweep control (MADDPG–VSC) algorithm has been developed, specifically tailored for the cooperative control of multiple variable-sweep wing UAVs. The algorithm employs an adaptive sweep angle control strategy to achieve efficient cooperative control in complex environments, significantly improving trajectory planning and obstacle avoidance capabilities.

(3) Construction of an adaptive optimization system for multi-objective reinforcement learning.

A multi-objective reward system has been designed to dynamically balance the performance requirements of multi-UAVs in complex tasks. This system ensures global optimization and rapid convergence, effectively avoiding local optima and enhancing the stability and flexibility of task execution.

In summary, while existing UAV formation control methods have established a foundation for multi-UAV systems, their limitations in handling continuous action spaces and

dynamic environments highlight the need for more advanced solutions. The MADDPG–VSC algorithm introduced in this study addresses these challenges, offering a novel approach to improving the performance of variable-sweep wing UAVs in complex, real-world scenarios. This study not only enhances the current methodologies but also provides a solid framework for future developments in multi-UAV cooperative control.

This study is structured as follows: Section 2 analyzes the dynamics of variable-sweep wing UAVs, including longitudinal dynamics and aerodynamic characteristics. Section 3 explores cooperative formation control using multi-agent reinforcement learning (MARL), focusing on the MADDPG–VSC algorithm's design, modeling, and simulation. Section 4 details experimental validation and performance analysis, including platform construction and application results. Finally, Section 5 summarizes the findings and suggests directions for future research.

2. Dynamics Analysis of the Variable-Sweep Wing UAV

The L-30A UAV, featuring a variable-sweep wing design, can adjust its sweep angles during flight to optimize aerodynamic efficiency and stability. This dynamic capability enhances flight performance but introduces significant nonlinearity and complexity due to changes in the center of mass and aerodynamic forces. To accurately simulate these complex dynamics, a multi-rigid-body dynamics model is essential. This model considers the interactions among various rigid bodies under different flight states, enabling precise simulation of the L-30A UAV's dynamic response.

2.1. Multi-Rigid-Body Dynamic Model

To simulate the complex dynamic behavior of the L-30A variable-sweep wing UAV during flight, a multi-rigid-body dynamics model based on the Newton–Euler equations is employed. This model accurately captures changes in the center of mass and aerodynamic characteristics caused by variations in sweep angles. Given the significant changes in the center of mass and inertia characteristics of the UAV's fuselage and wings during flight, the mass distribution across each rigid body is assumed to be uniform, and the UAV's geometric shape is considered symmetrical along the longitudinal axis. These assumptions provide the foundation for deriving the dynamic equations. In contrast to simplified models, this multi-rigid-body model dynamically adjusts to real-time variations in mass and inertia, enhancing control accuracy and stability across different flight conditions. Utilizing this approach, the dynamic relationships of the multi-rigid-body system are established, as illustrated in Figure 1.

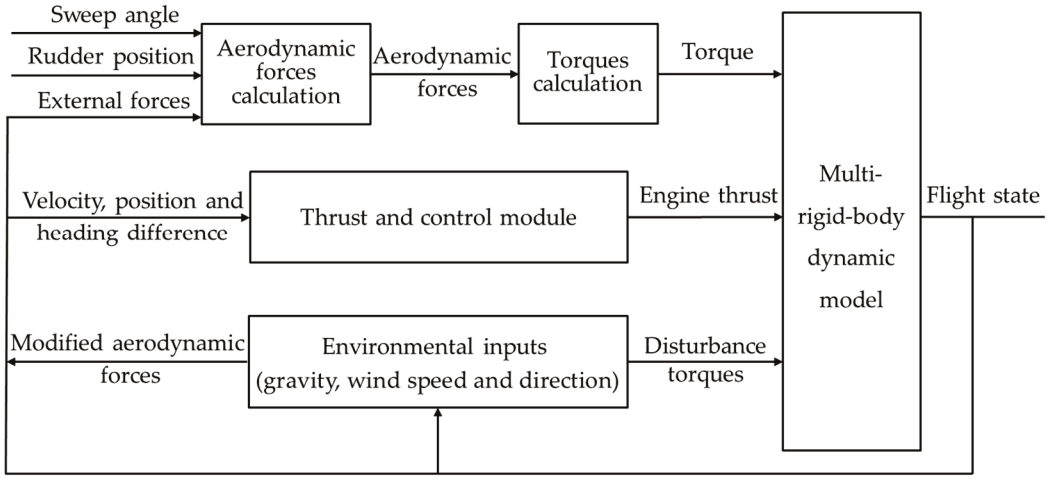
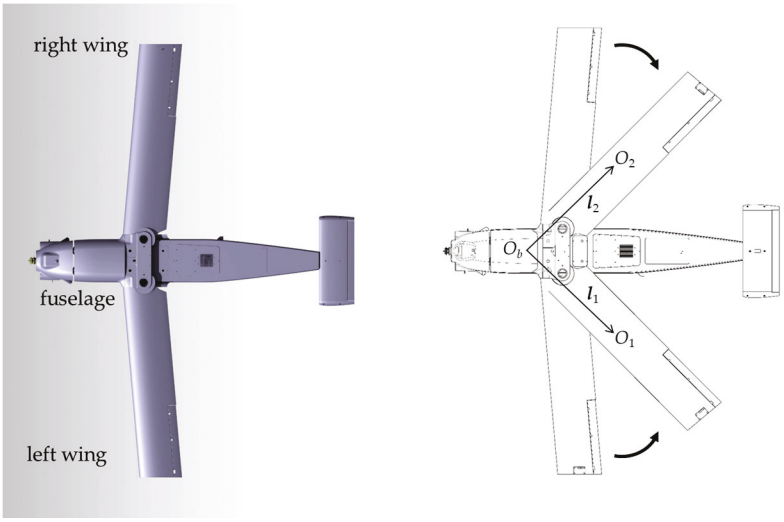


Figure 1. Schematic of dynamic relationships in a variable-sweep wing UAV.

(1) Expression of center of mass and velocity

The center of mass for each rigid body is defined, with O_i representing the center of mass, and $\boldsymbol{l}_i$ representing the position vector from the origin O_b of the fuselage coordinate system to O_i. The centers of mass for the left and right wings are denoted as O_1 and O_2, respectively, as shown in Figure 2. The relative velocity of O_i is given by

$$\boldsymbol{v}(\boldsymbol{p}_{O_i}) = \frac{d\boldsymbol{l}_i}{dt} = \boldsymbol{v}_i - \boldsymbol{v}_0 \quad (i = 1,2) \tag{1}$$



(**a**) (**b**)

Figure 2. Diagram of the variable-sweep wing UAV and mass centers. (**a**) Structural diagram of the L-30A variable-sweep wing UAV. (**b**) Schematic of mass centers and position vectors.

This equation describes the relative velocity of the wing's center of mass with respect to the fuselage. $\boldsymbol{v}_i$ is the velocity of the wing, and $\boldsymbol{v}_0$ is the velocity of the fuselage.

(2) Multi-rigid-body dynamics equation

The dynamics of each rigid body are defined. For each rigid body i, including the fuselage and two wings, the force equation is given by

$$\boldsymbol{F}_i = m_i \frac{dv_i}{dt} \quad (i = 0,1,2) \tag{2}$$

This equation describes the acceleration of the rigid body under the applied force. $\boldsymbol{F}_i$ represents the resultant force acting on the rigid body, m_i is the mass of the rigid body, and dv_i/dt is the acceleration of the rigid body.

(3) Total force and dynamic equilibrium of the system

By integrating the equations of motion for the centers of mass of both the fuselage and wings, the total dynamic equation for the system can be obtained, describing the overall dynamic behavior of the system during the morphing process, as follows:

$$\boldsymbol{F}_{\text{aero}} + \boldsymbol{G} + \boldsymbol{T} + \boldsymbol{F}_s = (m_0 + m_1 + m_2)\frac{dv_0}{dt} \tag{3}$$

This equation combines all forces acting on the system, including aerodynamic force $\boldsymbol{F}_{\text{aero}}$, gravity $\boldsymbol{G}$, thrust $\boldsymbol{T}$, and additional forces $\boldsymbol{F}_s$ generated by wing deformation, to calculate the system's acceleration dv_0/dt.

(4) Calculation of additional force

The additional force $\boldsymbol{F}_s$ generated during the wing deformation process reflects the impact of structural changes on the aerodynamic and inertial forces. The expression for $\boldsymbol{F}_s$ is given by

$$\boldsymbol{F}_s = \sum_{i=1}^{2} \boldsymbol{F}_{s_i} = \sum_{i=1}^{2} - m_i \frac{d\boldsymbol{v}\left(\boldsymbol{p}_{O_i}\right)}{dt} \tag{4}$$

(5) Simplification of forces and moment

To simplify the dynamic analysis, it is assumed that the centers of mass lie within the plane of the fuselage coordinate system with longitudinal symmetry, and that the dynamics are primarily influenced by longitudinal components. Consequently, the calculation of additional forces can be further simplified as follows:

$$\boldsymbol{F}_s = 2m_1 \left[-\ddot{l}_{1x} + \omega_y^2 l_{1x} + 2\omega_y \dot{l}_{1x} + \dot{\omega}_y l_{1x} \right] \tag{5}$$

where l_{1x} represents the position of the left swept wing, $\dot{l}_{1x}$ represents its rotational velocity, and $\ddot{l}_{1x}$ represents its rotational acceleration. ω_y represents the angular velocity of the centroid point of the deformable wing rotating about the origin point O_b of the fuselage coordinate system, and $\dot{\omega}_y$ represents the angular acceleration of the centroid point of the deformable wing rotating about the origin point O_b of the fuselage coordinate system. The factor of 2 accounts for the symmetrical contribution of the left and right wings, assuming they generate identical additional forces. The forces generated by the fuselage are considered negligible in this context due to the dominance of the wing dynamics in the overall behavior.

The simplified expression highlights the important longitudinal force components and their impact on dynamic behavior. The moment $\boldsymbol{H}_{o_b}$ about the origin O_b can be expressed as the integral of the cross product of position and velocity, as follows:

$$\boldsymbol{H}_{o_b i} \equiv \int (\boldsymbol{l}_i^* \times \boldsymbol{v}_i^*) dm_i^* \quad (i = 1,2) \Big) \tag{6}$$

where $\boldsymbol{l}_i^*$ represents the vector radius of O_b, and $\boldsymbol{v}_i^*$ represents the velocity in the inertial coordinate system.

The time derivative of the moment can be computed to reflect the dynamic changes in response to morphing actions, as follows:

$$\frac{d\boldsymbol{H}_{o_b i}}{dt} = \int (\boldsymbol{l}_i^* \times \frac{d\boldsymbol{v}_i^*}{dt}) dm_i^* + \int (\frac{d\boldsymbol{l}_i^*}{dt} \times \boldsymbol{v}_i^*) dm_i^* = M_{o_i} + m_i \dot{\boldsymbol{l}}_i \times \boldsymbol{v}_0 \tag{7}$$

where M_{o_i} represents the external torque applied to each wing around O_b, and $\dot{\boldsymbol{l}}_i$ represents the time derivative of the position vector of the wing.

(6) Dynamic equilibrium of the system

In a state of force and moment equilibrium, the longitudinal dynamic equations for the variable-sweep wing UAV can be expressed as follows:

$$\begin{cases} \boldsymbol{T}\cos\alpha - \boldsymbol{D} - \boldsymbol{G}\sin\vartheta + F_{p_x}\cos\alpha + F_{p_z}\sin\alpha = m\frac{d\boldsymbol{v}}{dt} \\ \boldsymbol{T}\sin\alpha + \boldsymbol{L} - \boldsymbol{G}\cos\vartheta + F_{p_x}\sin\alpha - F_{p_z}\cos\alpha = m\boldsymbol{v}\frac{d\vartheta}{dt} \\ (I_{1y} + I_{2y})\frac{d\omega_y}{dt} = M_{\mathrm{SG}} + M_{\mathrm{SD}} \end{cases} \tag{8}$$

where α represents the angle of attack, and ϑ represents the pitch angle. $\boldsymbol{D}$ represents the drag force acting on the UAV, defined as $\boldsymbol{D} = C_d q S_{\mathrm{ref}}$, where C_d is the drag coefficient, and q is the dynamic pressure, calculated as $q = \frac{1}{2}\rho v^2$, with ρ representing the air density and v indicating the velocity. S_{ref} is the reference area. $\boldsymbol{L}$ represents the lift force of the UAV, defined as $\boldsymbol{L} = C_l q S_{\mathrm{ref}}$, where C_l is the lift coefficient. I_{1y}, and I_{2y} represent the moments

of inertia along the y-axis for the left and right wings, respectively. M_{SG} represents the moment due to gravitational force along the radius vector, multiplied by the moment arm. M_{SD} represents the moment due to aerodynamic resistance.

(7) Relationship between sweep angle and moment

To specifically analyze the effect of the sweep angle on the system's dynamics, the relationship between the sweep angle and the geometric configuration of the wings can be expressed as

$$l_{1x} = a_p \sin \Lambda \tag{9}$$

This equation links the geometric configuration of the wings to the angular changes in the sweep angle Λ, providing a basis for calculating specific aerodynamic forces and moments. The term a_p is a constant related to the wing's shape.

(8) Linearization of the dynamic equation

To simplify the control analysis, the longitudinal dynamic equations of the system can be linearized as follows:

$$\dot{v} = \frac{0.5\rho v^2 S_{\text{ref}} C_t \cos\alpha}{m_i} - \frac{0.5\rho v^2 S_{\text{ref}} C_d}{m_i} - g\sin\vartheta + \frac{2m_1(-\ddot{l}_{1x} + \omega_y^2 l_{1x})\cos\alpha}{m_i} \tag{10}$$

where C_t is the thrust coefficient, and g represents the acceleration due to gravity.

This linearization is valid under the assumption of small angles, where the variations in angle of attack α and pitch angle ϑ are within a limited range. Such conditions are typical for steady-level flight and small perturbations around equilibrium. In practical applications, this approximation effectively simplifies the analysis and control of the system, while still capturing the essential dynamics. However, for scenarios involving large angular changes or highly dynamic maneuvers, the nonlinear effects could become significant and require a more comprehensive modeling approach.

2.2. Dynamic Characteristics Analysis of the L-30A UAV

To comprehensively understand the aerodynamic characteristics of the L-30A variable-sweep wing UAV, the multi-rigid-body dynamics model established earlier is used to analyze the aerodynamic forces and moments experienced by the L-30A under various flight states, including takeoff, cruising, maneuvering, and diving. This analysis aims to provide a foundation for optimizing control strategies.

(1) Connection between the model and aerodynamic analysis

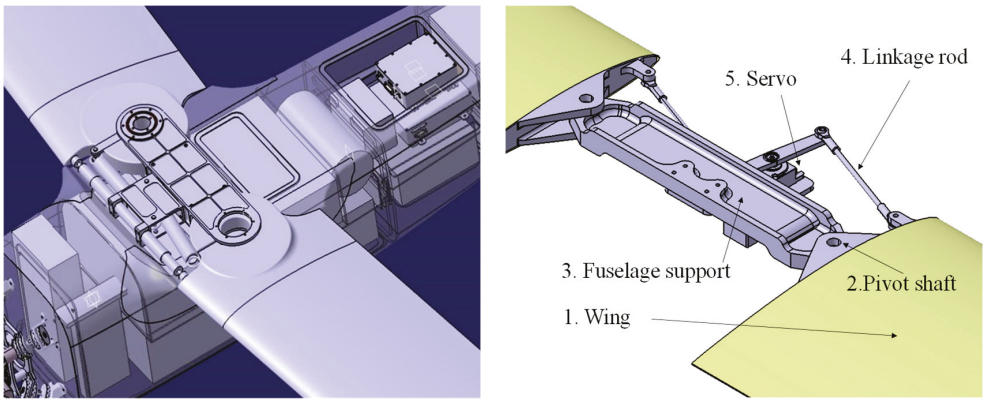
The multi-rigid-body dynamics model enables precise calculations of the aerodynamic forces and moments acting on the L-30A UAV. These calculations are crucial for understanding the UAV's performance during different flight states and for determining the most effective control strategies to optimize performance.

(2) Detailed analysis of dynamic characteristics

This study focuses on the L-30A variable-sweep wing UAV, an evolution of the L-30 fixed-wing UAV originally designed with a fixed 6° sweep angle. The L-30A introduces a critical rotating mechanism at the wing–fuselage junction, allowing for adjustment of the wing's sweep angle during flight, as shown in Figure 3. The key performance parameters of the L-30 series UAVs are compared in Table 1.

(3) Aerodynamic characteristics analysis

The aerodynamic characteristics of the L-30A UAV were analyzed across various flight states, including takeoff, cruising, maneuvering, and diving. To ensure optimal performance, the sweep angle of the variable-sweep wing was restricted to 16° during maneuvering and 60° during diving. The aerodynamic forces and moments calculated using the multi-rigid-body dynamics model, as detailed in Table 2, are critical for understanding the UAV's behavior during these states.



(**a**) (**b**)

Figure 3. Rotating mechanism at the wing–fuselage junction of the L-30A UAV. (**a**) Schematic of the rotation mechanism. (**b**) Schematic of the wing–fuselage junction detail.

Table 1. Flight performance parameters of L-30 series UAVs.

Parameters	L-30A Variable-Sweep Wing UAV	L-30 Fixed-Wing UAV
Takeoff weight	75 kg	70 kg
Service ceiling	6800 m	7000 m
Velocity	100–120 km/h (cruising) 180 km/h (maximum)	100–120 km/h (cruising) 150 km/h (maximum)
Endurance	4.8 h	5 h

Table 2. Geometric parameters of L-30A UAV's two wing configurations.

Parameters	Wing Configurations	
	16°	**60°**
Longitudinal Reference Length (m)	0.80408	1.98564
Chord Length (m)	0.33448	0.68220
Wingspan (m)	1.53164	0.69072

The geometric parameters of the L-30A UAV under these two sweep angles are summarized in Table 2. The table shows that at a 16° sweep angle, the wingspan is significantly larger and the chord length smaller, resulting in a longer longitudinal reference length. In contrast, at a 60° sweep angle, the wingspan is reduced, and the chord length is increased, which shortens the longitudinal reference length. These changes in geometry are crucial for adapting the UAV's aerodynamic performance to different flight states, particularly during high-speed maneuvers like diving.

Figure 4 illustrates the relationships between the lift coefficient, drag coefficient, and angle of attack. The lift coefficient shows a quasi-linear relationship with the angle of attack, while the drag coefficient exhibits a quasi-conic relationship. The analysis demonstrates that the maximum lift-to-drag ratio increases with greater sweep angles, highlighting the L-30A UAV's ability to meet diverse aerodynamic requirements across different flight states, particularly during the diving phase, where the highly swept wings provide enhanced efficiency.

(4) Comparative analysis of aerodynamic characteristics

A comparative analysis was conducted between the fixed-wing L-30 UAV with a 6° sweep angle and the variable-sweep configurations of the L-30A UAV during the maneuvering and diving phases. This comparison highlights the advantages of the variable-sweep wing design over the fixed-wing model in terms of performance and adaptability.

By integrating the multi-rigid-body dynamics model with the aerodynamic analysis, the control strategies for the L-30A UAV have been refined to optimize performance across

different flight states. The 16° sweep angle improves lift and stability during maneuvers, while the 60° sweep angle reduces drag in high-speed dives. This optimization enhances the UAV's adaptability and response in dynamic environments, providing robust support for control under challenging flight conditions. The findings highlight the clear advantages of the variable-sweep wing design over fixed-wing configurations.

The precise dynamic response of the variable-sweep wing UAV has been revealed through the above dynamic modeling. This model establishes a solid theoretical foundation for applying MARL algorithms in UAV formation control. Building on this foundation, the discussion will now focus on the design and application of the MADDPG–VSC algorithm to enhance the efficiency and stability of cooperative UAV control.

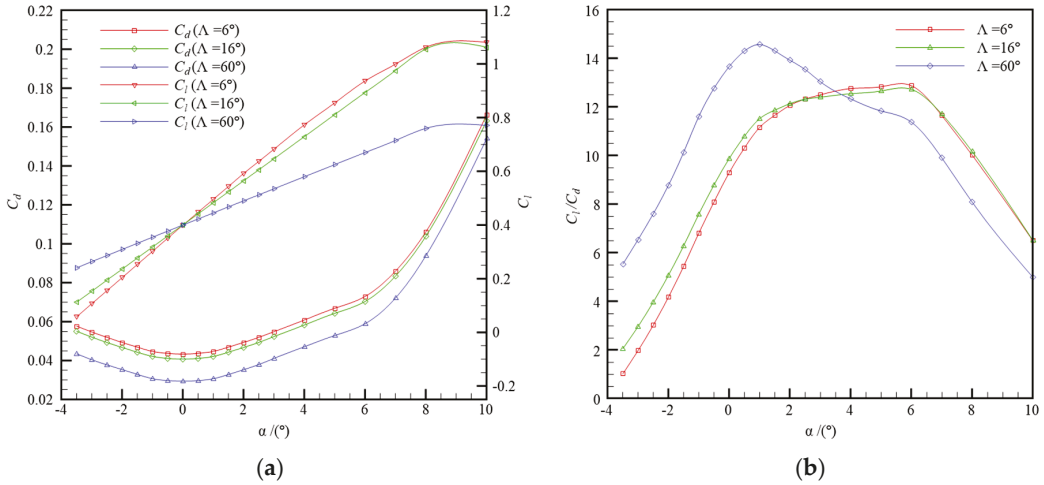


Figure 4. Aerodynamic characteristics of the L-30A variable-sweep wing UAV. (**a**) Lift and drag coefficients. (**b**) Lift-to-drag ratio.

3. Cooperative Control of Variable-Sweep Wing UAV via MADDPG

MARL has proven to be highly effective in the cooperative control of variable-sweep wing UAVs, particularly in achieving efficient formation control within complex environments. The primary challenge lies in managing continuous action spaces in dynamic flight states, which traditional reinforcement learning methods struggle to address. To overcome these challenges, the MADDPG–VSC algorithm has been specifically developed and optimized, taking full advantage of the unique characteristics of variable-sweep wing UAVs. This algorithm allows for dynamic adjustment of the sweep angle, thereby optimizing formation control and enhancing trajectory planning and obstacle avoidance capabilities, as illustrated in Figure 5.

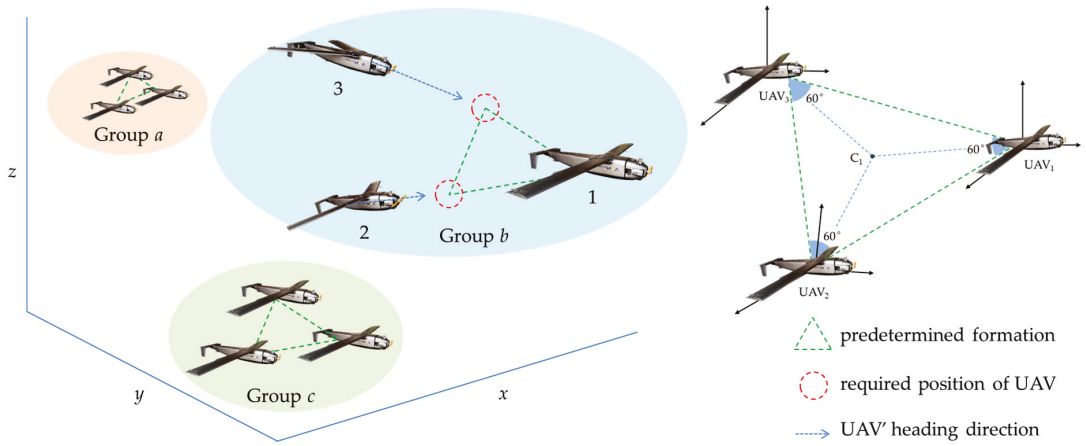


Figure 5. Schematic of cooperative formation control in variable-sweep wing UAV system.

3.1. Application of MARL in Variable-Sweep Wing UAV

In complex multi-agent systems, cooperative control is crucial for ensuring that individual agents, such as UAVs, can successfully accomplish their tasks together [18–21]. However, this process poses several challenges, particularly when managing continuous action spaces in dynamic flight states. The MADDPG–VSC algorithm builds upon the MARL framework, which can be represented as a high-dimensional tuple $< S, A_1, A_2, \ldots, A_N, R_1, R_2, \ldots, R_N, P_{dyn}, \gamma >$ [22–25]. In this framework, the state space S represents the key states of the UAVs, including position, velocity, and sweep angle; the action space A encompasses control inputs such as thrust, steering angle, and control voltage. The reward function R evaluates the contribution of each action to the success of the mission; the state transition dynamics P_{dyn} is defined as $P_{dyn} : S \times A \times S \to [0, 1]$, describing the probability of state changes; and the discount factor γ determines the current value of future rewards [26–28]. These elements together form the foundation of the MADDPG–VSC algorithm, providing the theoretical support needed for dynamic cooperative control of UAVs in complex environments.

The MADDPG–VSC algorithm incorporates state-value and action-value functions, alongside Q-learning and policy gradient mechanisms, to optimize UAV control in real time. Their specific implementations in the MADDPG–VSC algorithm are as follows:

(1) State-value function $V_i(s)$

The state-value function evaluates the total future reward that agent i may receive while in state s. For UAV formation control tasks, predicting the long-term effects of different flight trajectories is crucial. Through the state-value function, the MADDPG–VSC algorithm can assess the UAV's flight state in real-time, enabling the selection of the optimal flight strategy at each moment. The state-value function $V_i(s)$ is expressed as follows:

$$V_i(s) = E_i\left[r^i_{t+1} + \gamma V_i(s_{t+1})|s_t = s\right] \tag{11}$$

(2) Action-value function $Q_i(s, a)$

The action-value function evaluates the total reward that can be obtained by taking action a in state s. This allows the MADDPG–VSC algorithm to guide the UAVs in selecting the optimal control strategy in complex environments, ensuring formation stability and mission success. The action-value function $Q_i(s, a)$ is given by

$$Q_i(s, a) = E_i\left[r^i_{t+1} + \gamma Q_i(s_{t+1}, a_{t+1})|s_t = s, a_t = a\right] \tag{12}$$

(3) Q-learning update

The Q-learning update mechanism iteratively updates the Q-values to continuously optimize the agent's strategy, leading to convergence towards an optimal strategy. Through this mechanism, the MADDPG–VSC algorithm can dynamically adjust strategies during flight to adapt to changing environments and task demands. The Q-learning update mechanism is governed by the following equation:

$$Q_{t+1}(s_t, a_t) = Q(s_t, a_t) + \eta\left[r_{t+1} + \gamma max_{a_{t+1}} Q_{t+1}(s_{t+a}, a_{t+1}) - Q(s_t, a_t)\right] \tag{13}$$

(4) Policy gradient update

In dynamic environments, traditional Q-learning methods may struggle to handle continuous action spaces. To address this, the MADDPG–VSC algorithm employs a policy gradient update mechanism, which optimizes the policy parameters θ^μ. This mechanism guides UAVs in selecting the optimal sweep angle adjustment strategy, thereby improving overall task success. The policy gradient update is formulated as follows:

$$\nabla_{\theta^\mu} J \approx E\left[\nabla_{\theta^\mu} Q_\pi\left(s, a\middle|\theta^Q\right)\middle|s = s_t, a = \mu(s_t|\theta^\mu)\right] \cdot \nabla_{\theta^\mu} \mu(s|\theta^\mu)|s = s_t \tag{14}$$

Through these core mechanisms, including the state-value function, action-value function, Q-learning update mechanism, and policy gradient update mechanism, the MADDPG–VSC algorithm enables efficient cooperative control of UAVs in complex and dynamic environments. These mechanisms work together to ensure the stability and flexibility of the algorithm.

3.2. Optimization of the MADDPG–VSC Algorithm for Control Design

To address the limitations of traditional reinforcement learning algorithms in controlling variable-sweep wing UAVs, this study introduces the MADDPG–VSC algorithm. Unlike traditional algorithms that primarily target discrete action spaces, the MADDPG–VSC algorithm is specifically designed for continuous action spaces, enabling real-time adjustments of wing configurations according to flight states. This ensures the aerodynamic efficiency and stability of UAV formations.

3.2.1. Reward Function Design

The MADDPG–VSC algorithm's reward function balances multiple objectives, including sweep angle adjustment, formation maintenance, energy efficiency, and obstacle avoidance. Each component contributes to optimal UAV performance in dynamic environments.

(1) Sweep angle reward r_Λ

This reward is given when the algorithm selects the sweep angle Λ based on the current state s. It evaluates how closely the adjusted sweep angle matches the optimal sweep angle Λ_{opt}, thus quantifying the effectiveness of the adjustment in specific flight states. The reward function optimizes sweep angle adjustments for better aerodynamic efficiency and UAV formation stability. Typically, the reward r_Λ^+ is set to a positive value, such as 1 or 10, depending on the importance of maintaining optimal sweep angles. The parameter β_Λ is set to 0.1, allowing for a 10% deviation from the optimal sweep angle to ensure stability while optimizing performance.

$$r_\Lambda = \begin{cases} r_\Lambda^+ & if|\Lambda - \Lambda_{opt}| \leq \beta_\Lambda \\ 0 & otherwise \end{cases} \tag{15}$$

(2) Formation reward $r_{\text{formation}}$

Assuming that three UAVs form an equilateral triangle formation, the virtual center point C is the geometric center of the three UAVs and follows the designated flight trajectory. The formation reward is based on the deviation of each UAV from the center point C, with smaller deviations indicating a more stable formation and thus, yielding a higher reward. This calculation ensures that the formation remains tight and stable, which is critical for maintaining formation performance [29]. The value of $\alpha_{\text{formation}}$, which influences the importance of formation maintenance, is typically set to 10. This value balances the need for close formation with other mission objectives, such as energy efficiency and obstacle avoidance. The formation reward is then calculated as

$$r_{\text{formation}} = \alpha_{\text{formation}} \cdot \left(1 - \frac{d_{\text{avg}}}{d_{\max}}\right) \tag{16}$$

where d_{avg} represents the average deviation distance of the N UAVs from the center point C, calculated as $d_{\text{avg}} = \frac{1}{\text{N}} \sum_{i=1}^{\text{N}} \|p_i - p_C\|$. Here, p_i represents the position of each UAV, and p_C represents the position of the center point C. The term $d_{\max}$ denotes the maximum allowable deviation distance, beyond which the formation is considered unstable.

Further, the relative distance d between UAVs is defined as:

$$d = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2 + (z_{i+1} - z_i)^2} \tag{17}$$

where (x_i, y_i, z_i) and $(x_{i+1}, y_{i+1}, z_{i+1})$ represent the positions of two UAVs in the formation. Additionally, the relative distance d between UAVs is subject to the constraint $d \leq d_{\max}$, ensuring that the formation remains stable. If the relative distance d between UAVs exceeds $d_{\max}$, the formation is considered unstable, and the reward decreases accordingly.

(3) Energy consumption penalty r_{energy}

This penalty is incurred for unnecessary energy consumption, optimizing UAV endurance by minimizing energy wastage through thrust and flight trajectory optimization. The typical setting for α_{energy}, which discourages inefficient energy use, is 5, promoting a constant and efficient flight velocity. The energy consumption penalty is approximated by using the velocity Δv of the UAV, which is a proxy for energy usage. $|\Delta v|$ can be defined as

$$|\Delta v| = |v_t - v_{t-1}| \tag{18}$$

where v_t and v_{t-1} represent the magnitude of the UAV's velocity vector at the current time step t and the previous time step $t-1$, respectively. Thus, the energy consumption penalty is expressed as

$$r_{\text{energy}} = \alpha_{\text{energy}} \cdot |\Delta v| \tag{19}$$

(4) Collision avoidance penalty $r_{\text{collision}}$

The collision risk C_{risk} represents the likelihood of a collision based on the proximity to obstacles or other UAVs. It can be calculated using a function that evaluates the inverse distance, as follows:

$$\mathrm{C}_{\text{risk}} = \frac{1}{d + \epsilon} \tag{20}$$

where d is the relative distance between UAVs or between a UAV and an obstacle, and ϵ is a small positive constant used to prevent division by zero.

This penalty discourages potential collision events, encouraging UAVs to avoid obstacles and maintain safe distances in complex environments [30]. The value of $\alpha_{\text{collision}}$, which reflects the priority of collision avoidance, is typically set to 15. This value ensures that avoiding risks is prioritized, while maintaining a balance with other mission objectives. The collision avoidance penalty is then defined as

$$r_{\text{collision}} = \alpha_{\text{collision}} \cdot \mathrm{C}_{\text{risk}} \tag{21}$$

The overall reward function R is expressed as

$$R = r_{\Lambda} + r_{\text{formation}} - r_{\text{energy}} - r_{\text{collision}} \tag{22}$$

This reward function system effectively balances positive reinforcement and penalties, guiding the algorithm toward optimal solutions that meet mission requirements [31]. By optimizing these reward components, the algorithm enhances the overall performance of UAV formations, ensuring that the best control strategies are learned and applied.

3.2.2. Exploration and Learning Efficiency

To ensure adequate exploration in continuous action spaces, the MADDPG–VSC algorithm introduces random noise ξ to action exploration. This noise introduces variability in action selection, preventing the algorithm from converging prematurely to suboptimal policies. The noise parameter c is dynamically adjusted during the training process to balance exploration and exploitation. Initially, c is kept high to encourage extensive exploration, but it gradually decreases as the algorithm converges to a more stable policy, as follows:

$$u' = u + \xi \tag{23}$$

where ξ follows a normal distribution $N(u, c)$, with mean u and variance c. The noise parameter c is defined as

$$c = \begin{cases} c & \text{if episode} \leq 50 \\ 0.995c & \text{if episode} > 50 \end{cases} \tag{24}$$

To enhance learning efficiency, the sweep angle Λ and other state values are normalized in the input layer, yielding dimensionless state value $\tilde{\Lambda}$. This normalization is defined as

$$\tilde{\Lambda} = \frac{\Lambda - \Lambda_{\min}}{\Lambda_{\min} - \Lambda_{\max}} \tag{25}$$

where $\Lambda_{\min}$ represents the minimum sweep angle during UAV flight, and $\Lambda_{\max}$ represents the maximum sweep angle. Normalization helps in scaling state values to a uniform range, improving the algorithm's training efficiency and stability by ensuring that all input features are on a comparable scale, thus contributing equally to the model's learning process.

3.2.3. Convergence and Stability

The MADDPG–VSC algorithm significantly improves training stability through the introduction of a soft update mechanism and the use of target networks. Given that the environment model and the state transition probability P are unknown, these aspects present challenges in reinforcement learning, particularly when dealing with complex and dynamic environments.

To address these challenges, the MADDPG-VSC algorithm utilizes a soft update approach, which gradually integrates new information into the model, ensuring that updates are smooth and that they do not destabilize the learning process. This method involves the construction of a policy network $\mu(s|\theta^{\mu})$, a Q network $Q(s, a|\theta^{Q})$, and their respective target networks $\mu'\left(s\middle|\theta^{\mu'}\right)$ and $Q'\left(s, a\middle|\theta^{Q'}\right)$. The soft update mechanism is expressed as

$$\begin{cases} \theta^{\mu'} = \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \\ \theta^{Q'} = \tau\theta^{Q} + (1-\tau)\theta^{Q'} \end{cases} \tag{26}$$

This gradual update method mitigates risks such as gradient explosion or vanishing, stabilizing the training process of the MADDPG–VSC algorithm. It allows the algorithm to progressively approach the optimal policy with enhanced stability, making it particularly effective in the complex and continuously changing environments encountered by variable-sweep wing UAVs.

3.2.4. Adaptive Sweep Angle Control Strategy

The MADDPG–VSC algorithm employs an adaptive sweep angle control strategy, dynamically adjusting the sweep angle in response to real-time flight parameters. This strategy is critical for optimizing UAV formation control, as it allows for adjustments that enhance aerodynamic efficiency and overall mission success. By dynamically responding to changes in velocity, altitude, and other flight conditions, the algorithm ensures that UAV formations maintain optimal configurations throughout various flight states.

This adaptive approach not only improves trajectory planning and obstacle avoidance but also significantly enhances overall UAV performance in diverse and challenging task scenarios. By integrating these techniques, the MADDPG–VSC algorithm provides a robust and flexible solution for cooperative UAV formation control in complex environments.

3.2.5. Flight States of UAV and Control Strategies

In the cooperative control of variable-sweep wing UAVs, the MADDPG–VSC algorithm dynamically adjusts control strategies based on real-time flight states, including velocity, attitude, and sweep angle [32]. These adjustments are crucial for maintaining

formation integrity and optimizing overall mission performance [33,34]. The control strategies are specifically tailored to each flight state, ensuring that UAVs can adapt to changing conditions and achieve precise formation assembly and reconfiguration when necessary. Table 3 presents the configurations for different flight states of the L-30A UAV, which are integral to executing cooperative maneuvers within the defined mission parameters.

Table 3. Flight states and sweep angle configurations of the L-30A UAV.

Flight States	Flight Velocity (km/h)	Flight Altitude (m)	Sweep Angle (°)
Takeoff	0–60	0–500	0–10
Climb	60–100	500–1000	10–20
Cruise	120–140	1000–2000	15–25
Maneuver	100–140	500–1500	20–35
Dive	120–160	0–1000	40–50
Vertical Attack	160–180	0–500	50–60
Landing	0–60	0–500	0–10

During the UAV formation process, specific control strategies are required for each flight state to ensure cooperation and stability. During takeoff, the primary focus is on generating sufficient lift with optimized thrust, while maintaining a minimal sweep angle for stability. This state is crucial for precise cooperation during formation assembly. As UAVs enter the climb state, the sweep angle is gradually increased to reduce drag, optimizing energy use while maintaining the integrity of the formation during the ascent.

In the cruise state, the UAVs stabilize with a moderate sweep angle, balancing velocity and fuel efficiency. Steady thrust ensures that the formation remains intact over long distances, allowing for minor adjustments when needed. Maneuvering demands rapid adjustments in rudder control and sweep angle, enabling UAVs to quickly respond to environmental changes and reconfigure the formation as required.

During dives, the UAVs adjust the sweep angle to reduce lift and increase velocity for a controlled descent. In the vertical attack state, UAVs adopt a high sweep angle to minimize drag and increase velocity, ensuring precision targeting while maintaining formation integrity. Finally, during landing, UAVs reduce velocity and lower the sweep angle to ensure stability and safety, with precise thrust and rudder control to facilitate a cooperative landing sequence for the entire formation.

These strategies ensure that UAVs can effectively adapt to various flight states by selecting and executing appropriate actions within the defined state space $S = \{x, y, z, v, \dot{v}, \Lambda\}$, where each element corresponds to critical state variables such as position, velocity, acceleration, and sweep angle. The ability to dynamically adjust these control inputs in response to specific flight states is crucial for maintaining formation integrity and achieving optimal performance across different operational states.

3.3. Structure and Training Process of the MADDPG–VSC Algorithm

3.3.1. Multi-Agent Algorithm Model Structure

The MADDPG–VSC algorithm is based on the DDPG framework and employs multi-agent DRL with an actor–critic architecture [35–37]. It is specifically designed to address the cooperative control challenges in variable-sweep wing UAVs operating in multi-agent environments. A key innovation of the algorithm lies in the introduction of an adaptive sweep angle control strategy, which enables precise optimization of continuous action spaces, allowing dynamic adjustments to formation strategies to meet complex mission requirements. Figure 6 illustrates the overall structure of the MADDPG–VSC algorithm, highlighting the flow of information and the interactions between various components during the training process. Each DDPG agent, such as a variable-sweep wing UAV, is equipped with its own actor and critic networks. The critic network in the MADDPG–VSC algorithm not only evaluates the state-action pairs of individual agents but also integrates

policy information from other agents, forming a global perspective that guides overall formation decisions. This sharing and cooperation of information among multi-agent is crucial for the algorithm to effectively handle complex tasks.

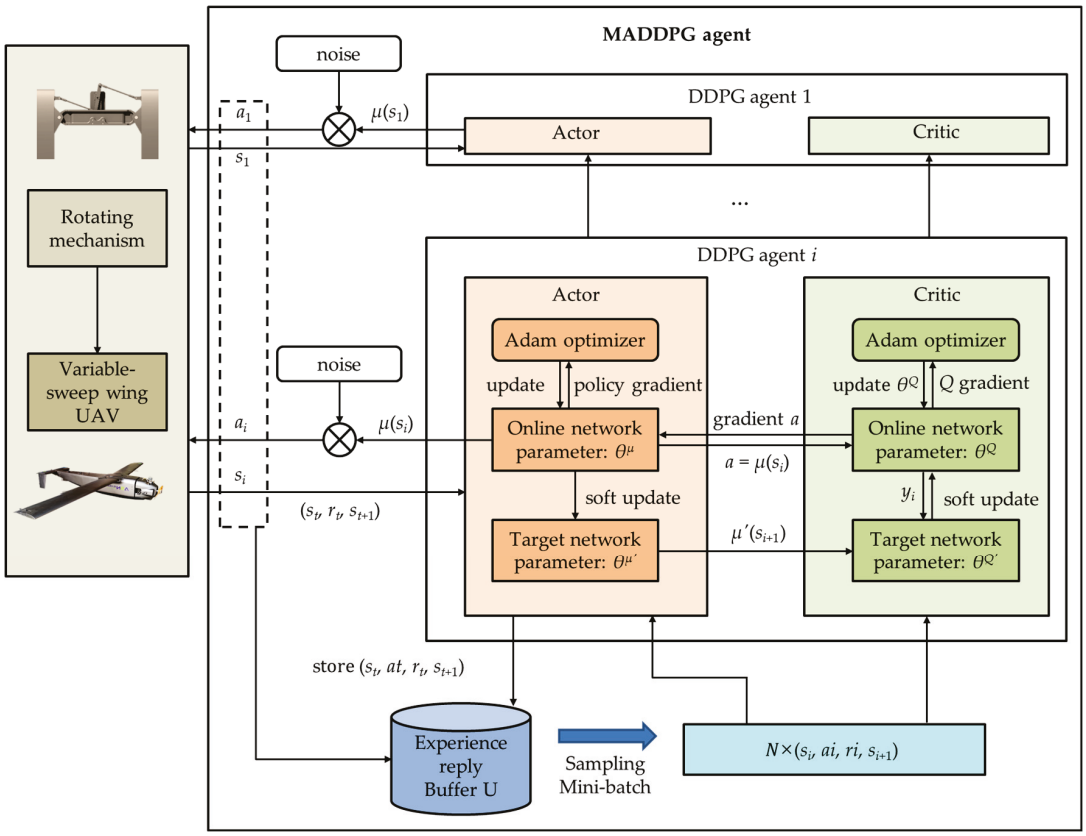


Figure 6. Structure of the MADDPG–VSC algorithm model.

The variable-sweep wing UAV, through the rotating mechanism depicted in the figure, demonstrates how the adaptive sweep angle control strategy dynamically adjusts the flight control parameters through real-time interactions with the environment. The actor network selects optimal actions based on the current state, while stochastic noise is introduced to facilitate policy exploration, and the critic network evaluates these actions. The MADDPG–VSC algorithm leverages actor–critic networks to adapt to complex environments, optimizing UAV formations and reducing task uncertainties. Moreover, the algorithm enhances training stability through the use of an experience replay buffer and a soft update mechanism, mitigating the risks of overfitting and unstable convergence. During each training cycle, both the critic and actor networks progressively optimize their policies, and soft updates gradually align the target networks with the current networks, ensuring efficient decision making and task execution in multi-agent, complex environments.

3.3.2. Training Process

During the training process of the MADDPG–VSC algorithm, agents continuously optimize their control strategies through interaction with the environment and other agents. The following steps outline the key components of the training process:

(1) Initialization.

The critic network parameters θ^Q and actor network parameters θ^μ for each agent are initialized. Additionally, the target network parameters $\theta^{Q'}$ and $\theta^{\mu'}$ are initialized, and the replay buffer U is set up for each agent.

(2) Policy selection.

At each time step t, agents select an action a_t based on its current state s_t using the policy $\mu(s_t|\theta^\mu)$. To encourage exploration within the continuous action space, stochastic noise ξ is added to the action, generating $a' = a + \xi$.

(3) Experience storage.

After executing the action, each agent receives a reward r_t and observes the next state s_{t+1}. The transition (s_t, a_t, r_t, s_{t+1}) is stored in the replay buffer U.

(4) Parameter updates.

When the replay buffer U reaches capacity, mini-batches of transitions are sampled for network updates. The target value $y_i = r_i + \gamma Q'\left(s_{i+1}, \mu'\left(s_{i+1}\middle|\theta^{\mu'}\right)|\theta^{Q'}\right)$ is computed, and the loss function is minimized to update the critic network parameters θ^Q. The actor network parameters θ^μ are updated through policy gradient backpropagation.

(5) Target network update.

After the critic and actor networks are updated, the target networks are softly updated using the following equations: $\theta^{\mu'} = \tau\theta^\mu + (1-\tau)\theta^{\mu'}$ and $\theta^{Q'} = \tau\theta^Q + (1-\tau)\theta^{Q'}$.

(6) Repeat training.

These steps are repeated throughout the training process until the specified number of episodes is completed or the algorithm converges, ultimately yielding a stable optimal policy. By providing a clear advantage in dynamic and unpredictable scenarios, the MADDPG–VSC algorithm enhances real-time trajectory planning and obstacle avoidance, where traditional algorithms often struggle. By incorporating multi-agent strategies, experience replay, and target networks, the training process is designed to enhance both stability and performance. As the critic and actor networks converge, the algorithm learns optimal control policies that adjust sweep angles and flight trajectories in response to changing environments. This ensures that UAV formations maintain stability and efficiency, enabling precise coordination and effective task execution, even in complex mission scenarios.

3.4. Simulation Verification and Performance Analysis of the MADDPG–VSC

3.4.1. Simulation Environment and Parameter Setting

The simulation environment effectively tests the MADDPG–VSC algorithm's capabilities in obstacle avoidance and cooperative formation control. The simulation environment is configured within a 1000 m × 1000 m area with a 2000 m altitude ceiling, creating a realistic and challenging space for testing these capabilities. Within this environment, three variable-sweep wing UAVs are initially positioned at specific locations and are tasked with navigating towards a stationary target while successfully avoiding seven strategically placed obstacles. This setup rigorously tests the effectiveness of the UAVs' control policies, particularly focusing on optimal sweep angle adjustments and coordinated movements necessary for obstacle avoidance and target acquisition. To evaluate the algorithm's robustness, two simulation scenarios were designed: a simple scenario with a regular quadrilateral flight path and no obstacles, serving as a baseline for basic coordination and trajectory tracking; and a complex scenario with a polygonal flight path, where UAVs must avoid obstacles, including simulated radar detection zones. These setups provide varying trajectory complexities to enhance the model's adaptability to environments significantly different from the training conditions. In addition to convex obstacles, future simulations will include randomly generated non-convex obstacles to better assess the algorithm's robustness and adaptability in dynamic environments. Figure 7 illustrates the detailed layout of this simulation environment, including the positions of the UAVs, the targets, and the radars, thereby highlighting the complex dynamics and challenges encountered during the simulation process.

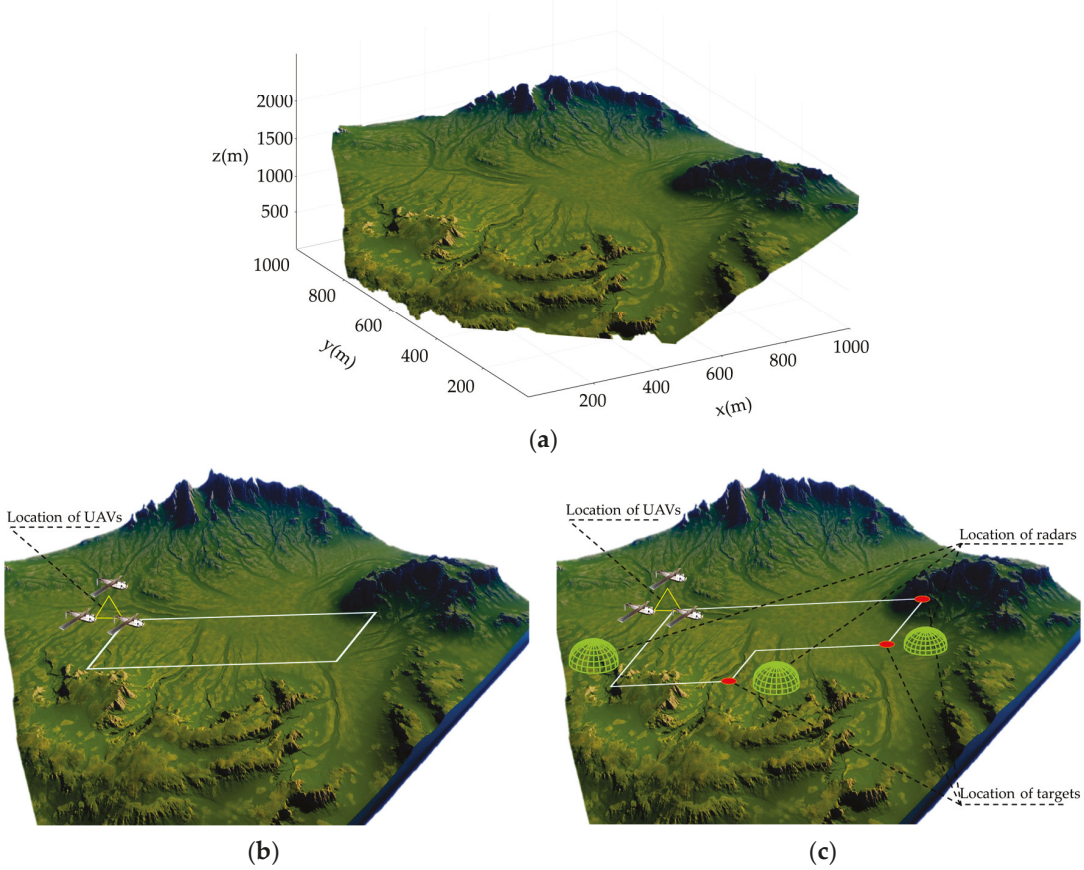


Figure 7. Schematic representation of the simulation environment. (**a**) Terrain schematic; (**b**) simple scenario; (**c**) complex scenario.

The MADDPG–VSC algorithm is implemented in Python using the TensorFlow 1.9 deep learning framework. The computational platform comprises an Intel E5-2630v4 @ 2.20 GHz CPU and 16GB of memory, ensuring sufficient resources for efficient training and execution. Training is conducted in a closed-loop manner, integrating both exploratory and high-quality experiences to enhance learning efficiency. Environmental data combined with the UAVs' state information guide the determination of control variables according to the evolving control policy. This policy is continuously refined through feedback from the environment, enabling progressive improvement in the UAVs' performance over time.

The simulation environment and parameter settings establish a robust foundation for evaluating the MADDPG–VSC algorithm's capabilities in regards to both obstacle avoidance and cooperative formation control tasks. Table 4 outlines the key hyperparameters used during training, ensuring clarity regarding the configuration and methodological considerations. This setup ensures a comprehensive and realistic assessment of the algorithm's performance.

Table 4. Configurations of hyperparameters.

Parameters	Values
Max Episodes	3000
Maximum Number of Steps T	200
Discount Factor γ	0.2
Critic Learning Rate η_1	0.002
Size of Buffer U	5000
Batch Size	32
Actor Learning Rate η_2	0.001

3.4.2. Algorithmic Performance Evaluation

The UAV trajectories, shown in Figure 8, compare the performance of the MADDPG, soft actor–critic (SAC) and MADDPG–VSC algorithms. MADDPG, known for its effectiveness in multi-agent reinforcement learning tasks, particularly in handling continuous action spaces, has been widely adopted in UAV formation control. SAC offers a robust alternative with its stochastic policy, effectively balancing exploration and exploitation, and is particularly known for its quick convergence and stable performance in continuous action tasks. Both MADDPG and SAC successfully guide the three variable-sweep wing UAVs toward their target, navigating obstacles. However, MADDPG–VSC shows distinct advantages in more dynamic scenarios. The presence of obstacles necessitates longer trajectories and the application of a deformation acceleration strategy. While all UAVs reached their destination using MADDPG and SAC, MADDPG shows a notable trajectory deviation for UAV_2 when encountering an obstacle, bringing it closer to the obstacle than intended. In comparison, SAC exhibits more precise performance than MADDPG in early training phases but struggles to maintain accuracy in more complex, obstacle-rich environments. In contrast, MADDPG–VSC demonstrates only minor deviations, resulting in a more accurate and efficient flight trajectory. This highlights MADDPG–VSC's superior precision and effectiveness in highly dynamic environments, particularly in managing precise navigation and obstacle avoidance, thus outperforming SAC in later stages of training and complex scenarios.

The results from Tables 5–7 demonstrate that MADDPG–VSC shows a clear advantage in regards to stability, as reflected in both maximum and total deviations. For instance, MADDPG–VSC's maximum deviation is 18.17 m, compared to MADDPG's 24.06 m and SAC's 19.00 m. Its total deviations of 11.83 m and 10.55 m are also smaller than those of SAC and MADDPG. This indicates that in multi-UAV cooperative tasks and complex environments, MADDPG–VSC achieves better trajectory control and reduced deviations. Its formation control capability and stability, particularly in dynamic scenarios, are clearly more reliable.

Table 5. UAV_1 absolute position error.

Algorithm	Maximum Deviation (m)	Average Deviation (m)	Median Error (m)	Total Deviation (m)
MADDPG	24.06	11.13	6.04	25.31
MADDPG–VSC	18.17	8.84	5.84	21.50
SAC	19.00	9.00	7.00	22.00

Table 6. UAV_2 absolute position error.

Algorithm	Maximum Deviation (m)	Average Deviation (m)	Median Error (m)	Total Deviation (m)
MADDPG	13.88	4.07	7.78	16.43
MADDPG–VSC	9.75	3.12	1.69	11.83
SAC	10.00	4.00	3.50	12.50

Table 7. UAV_3 absolute position error.

Algorithm	Maximum Deviation (m)	Average Deviation (m)	Median Error (m)	Total Deviation (m)
MADDPG	12.38	3.91	2.10	14.66
MADDPG-VSC	8.79	2.55	1.57	10.55
SAC	9.05	3.15	2.50	12.50

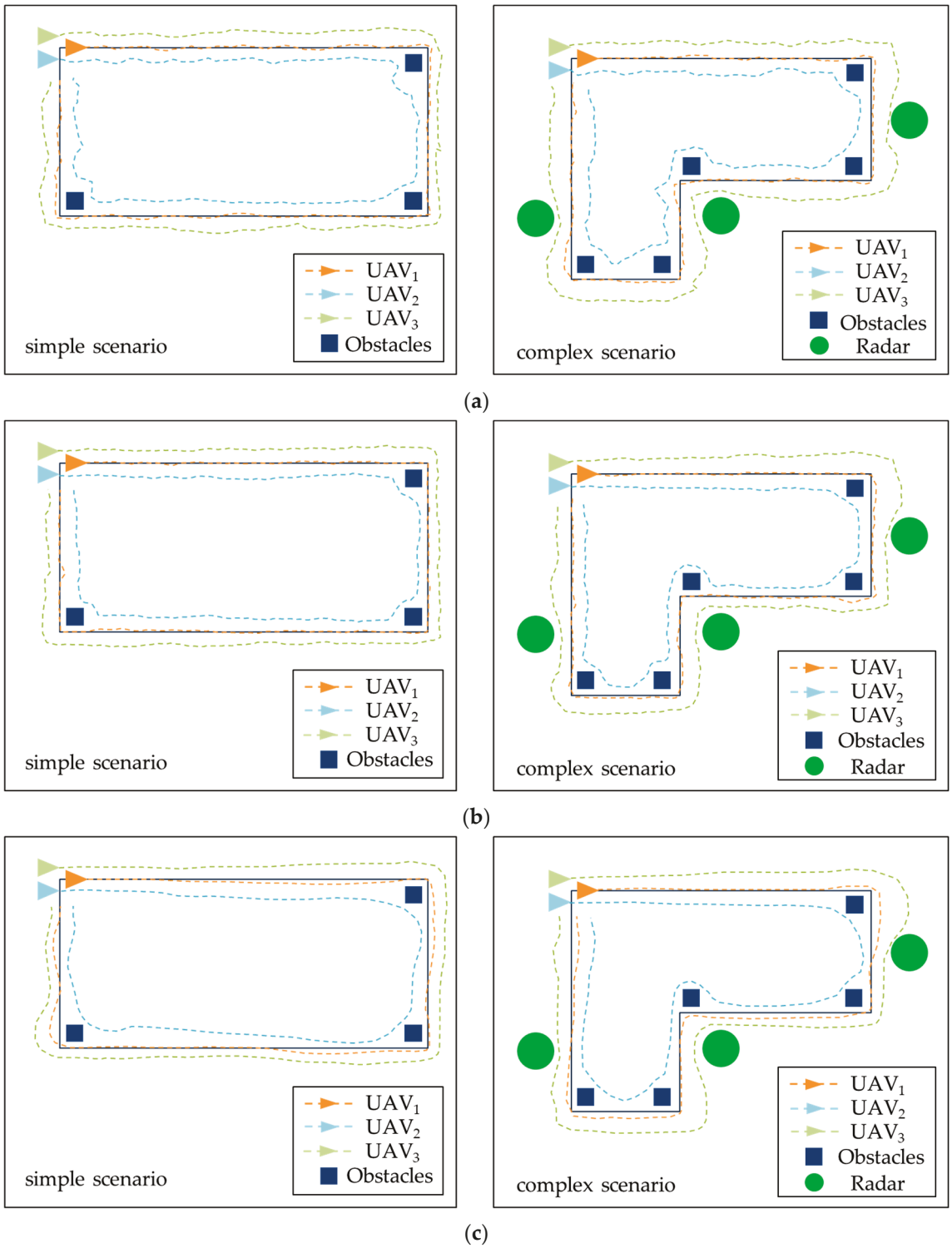


Figure 8. UAV trajectories using MADDPG, MADDPG–VSC, and SAC algorithms. (**a**) MADDPG algorithm; (**b**) MADDPG–VSC algorithm; (**c**) SAC algorithm.

Trajectory accuracy, reflected through average and median deviations, further emphasizes MADDPG–VSC's consistent performance. In Table 5, MADDPG–VSC's average deviation is 8.84 m, compared to MADDPG's 11.13 m and SAC's 9.00 m. Although SAC performs similarly to MADDPG–VSC in simpler environments, MADDPG–VSC excels in more complex scenarios. Similarly, in Table 7, MADDPG–VSC's average deviation is 2.55 m, outperforming MADDPG's 3.91 m and SAC's 3.15 m, indicating that MADDPG–VSC offers

more precise trajectory control, reducing deviations and keeping UAVs on their intended paths in complex tasks.

Additionally, the adaptive multi-objective reward system dynamically balances task priorities and optimizes UAV coordination in real-time, ensuring global optimization and preventing local optima, which is essential for maintaining stability across complex missions. In summary, MADDPG–VSC demonstrates clear advantages in regards to stability and trajectory accuracy when compared with both MADDPG and SAC. Although SAC occasionally performs similarly to MADDPG–VSC in simpler scenarios, it generally falls short in precision and stability in more complex, dynamic environments. These attributes make MADDPG–VSC more reliable for multi-UAV cooperative tasks, particularly in achieving precise navigation and formation control in challenging environments.

3.4.3. Comparative Analysis of Learning Efficiency and Convergence

To evaluate the training efficacy of the MADDPG–VSC algorithm, both MADDPG and MADDPG–VSC models were trained and compared. Figure 9 shows the reward curves, illustrating their performance and learning progress. In the complex simulation environment with multiple targets and obstacles, both models stabilized after around 3000 training episodes, indicating effective learning. However, stochastic noise during training caused some variability, seen as oscillations in the reward curves, which is common in reinforcement learning.

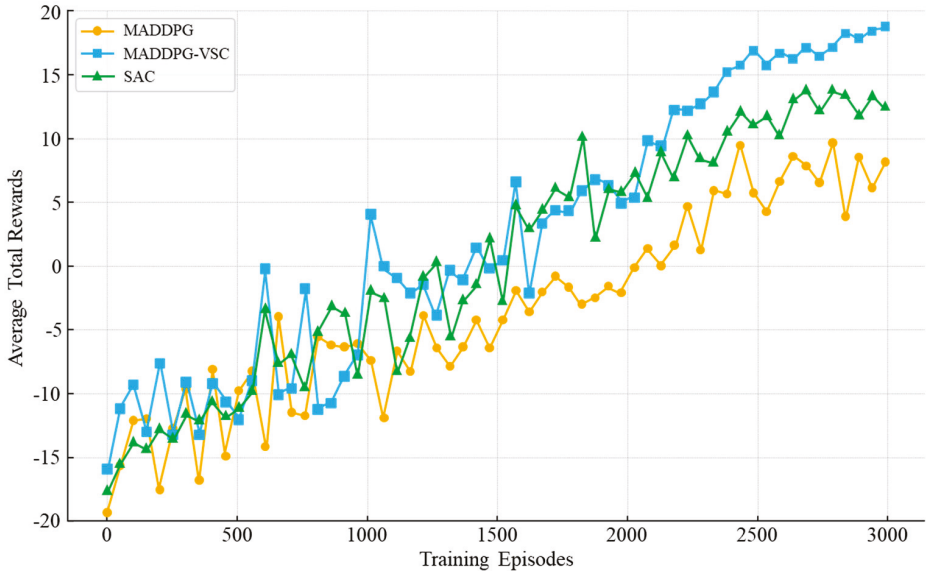


Figure 9. Training reward curves for MADDPG, MADDPG–VSC, and SAC algorithms.

Both algorithms stabilized after 2500 episodes, but the MADDPG–VSC algorithm achieved stability earlier, consistently yielding higher rewards throughout. This indicates a higher learning efficiency in handling tasks. The MADDPG algorithm exhibited significant fluctuations in later stages, particularly after approximately 1500 episodes, where reward values showed a slight decline. This suggests potential overfitting or instability issues during prolonged training. Although the algorithm eventually converged, the fluctuations indicate weaker adaptability and consistency when faced with complex tasks. In contrast, the MADDPG–VSC algorithm not only converged faster in the early training phase but also maintained a more stable reward value, demonstrating its advantages in multi-UAV formation and obstacle avoidance tasks. By incorporating adaptive sweep angle control, this algorithm significantly improved the stability and accuracy of task execution, particularly excelling in static complex environments.

To evaluate the performance of the MADDPG and MADDPG–VSC algorithms, their Q-value and loss value trends were compared over 3000 training episodes. Figure 10 shows

the results, providing insight into the learning progress of both models during multi-UAV formation control and obstacle avoidance tasks. The Q-value reflects the prediction of cumulative rewards, while the loss value indicates the prediction accuracy. Both models were tested under the same complex environment with multiple targets and obstacles.

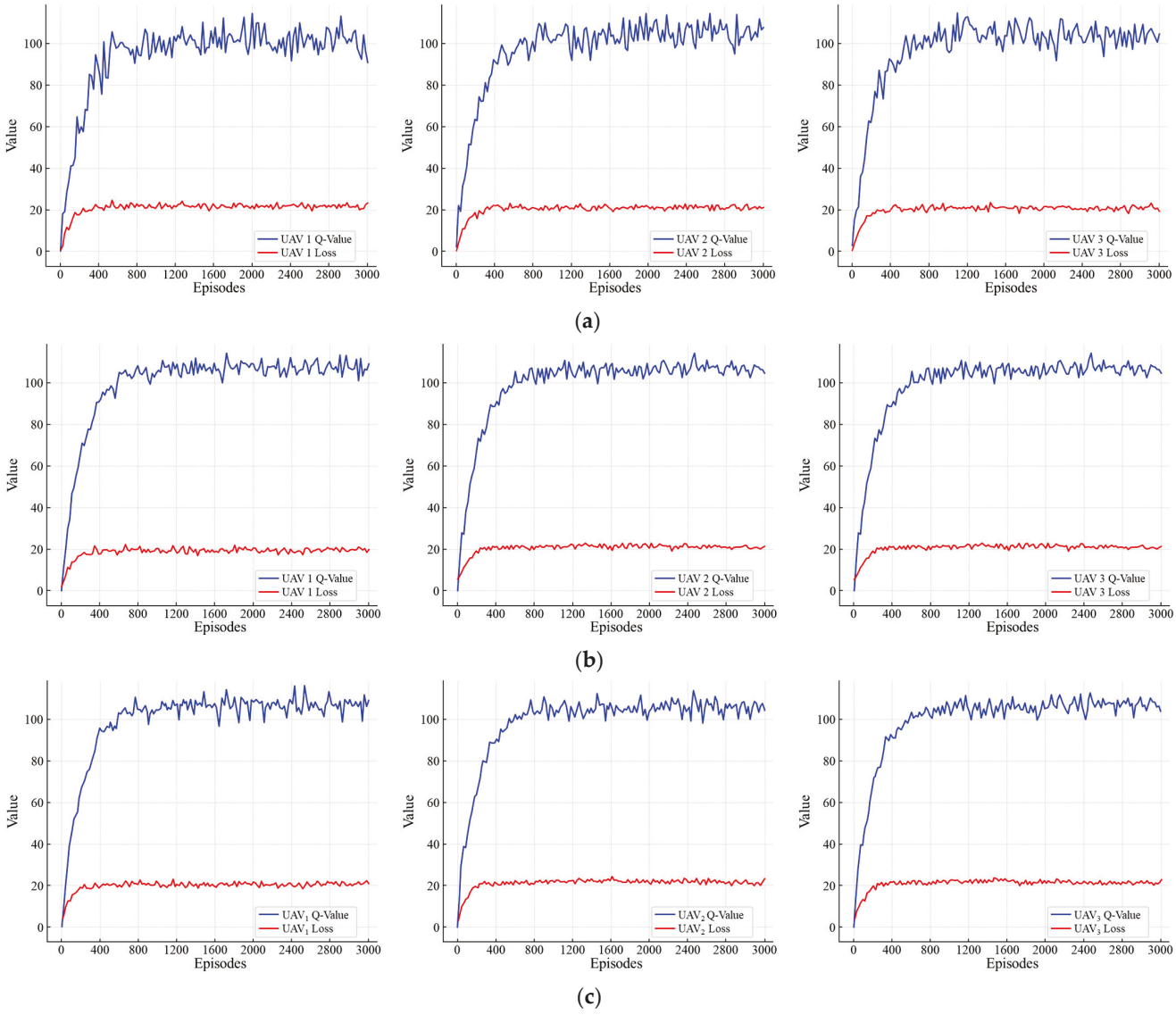


Figure 10. Network parameter variations using MADDPG, MADDPG–VSC, and SAC algorithms. (**a**) MADDPG algorithm; (**b**) MADDPG–VSC algorithm; (**c**) SAC algorithm.

During training, both algorithms initially showed a sharp increase in Q-values, indicating effective early learning. However, the MADDPG algorithm experienced a decline in Q-values after around 500 episodes, suggesting potential overfitting or instability. This decline indicates that MADDPG struggled to maintain consistent performance in more complex scenarios. Although it eventually converged, the fluctuations suggest weaker adaptability. In contrast, the MADDPG–VSC algorithm maintained a stable Q-value trend throughout training. Its integration of adaptive sweep angle control helped optimize performance, allowing the algorithm to achieve higher and more stable Q-values. This stability demonstrates its improved efficiency in regards to multi-UAV coordination and obstacle avoidance.

The loss value trends further support these findings. While both algorithms reduced their loss values over time, the MADDPG algorithm showed more variability, indicating less stable

learning. On the other hand, the MADDPG–VSC algorithm exhibited more consistent and lower loss values, reflecting higher prediction accuracy and better learning stability.

In summary, the MADDPG–VSC algorithm demonstrated greater stability and faster convergence, making it better suited for controlling variable-sweep wing UAVs in complex environments. The improvements in reward stability and loss reduction highlight its ability to handle dynamic and uncertain conditions, such as cooperative formation control and obstacle avoidance.

4. Flight Validation and Performance Analysis

This study constructed a flight validation platform using L-30A variable-sweep wing UAVs and conducted a series of formation flight experiments to evaluate the performance of the MADDPG–VSC algorithm, focusing on its role in cooperative control. The validation process includes the setup of the experimental platform, the execution of formation flights, and a detailed review of the experimental results.

4.1. Experimental Platform Overview

The validation of the MADDPG–VSC algorithm was conducted through a flight validation platform composed of L-30A variable-sweep wing UAVs. This platform was designed to test the effectiveness of the algorithm in cooperative control tasks. The platform was configured to balance hardware performance with algorithmic demands, ensuring reliable and precise flight control.

4.1.1. L-30A UAV System

The L-30A UAV system, as shown in Figure 11, consists of variable-sweep wing UAVs, a ground control station (GCS), and a data link, all designed to execute cooperative multi-UAV tasks. The L-30A UAV dynamically adjusts its sweep angle to optimize aerodynamic performance, while its integrated sensors, including an inertial measurement unit (IMU), a global positioning system (GPS), and high-resolution cameras, provide real-time data for position, velocity, attitude angles, and sweep angle. The system uses a data link to enable high-speed, low-latency communication between the UAVs and the GCS, facilitating the transmission of flight status, sensor data, and control commands. The GCS acts as the command center, receiving and analyzing UAV data in real time, allowing for mission control and flight adjustments, while the UAV formation maintains coordination autonomously to ensure precise execution of cooperative tasks.

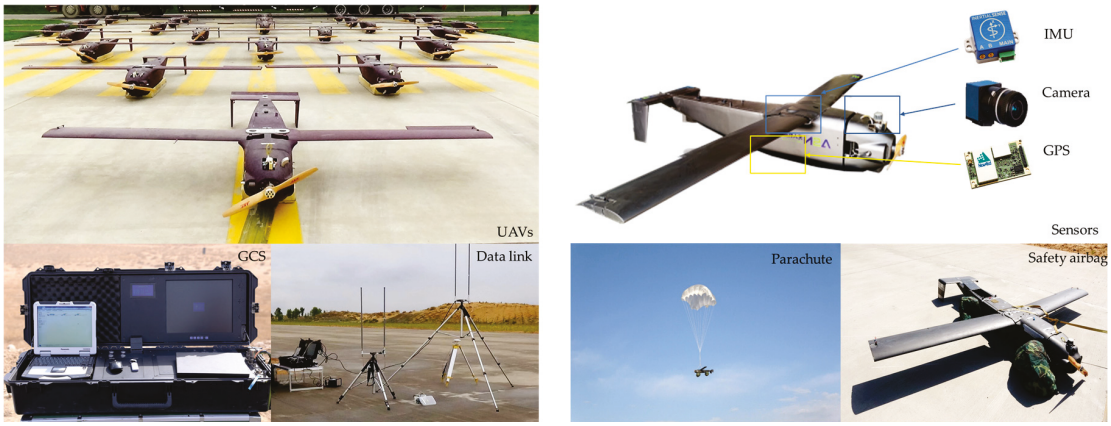
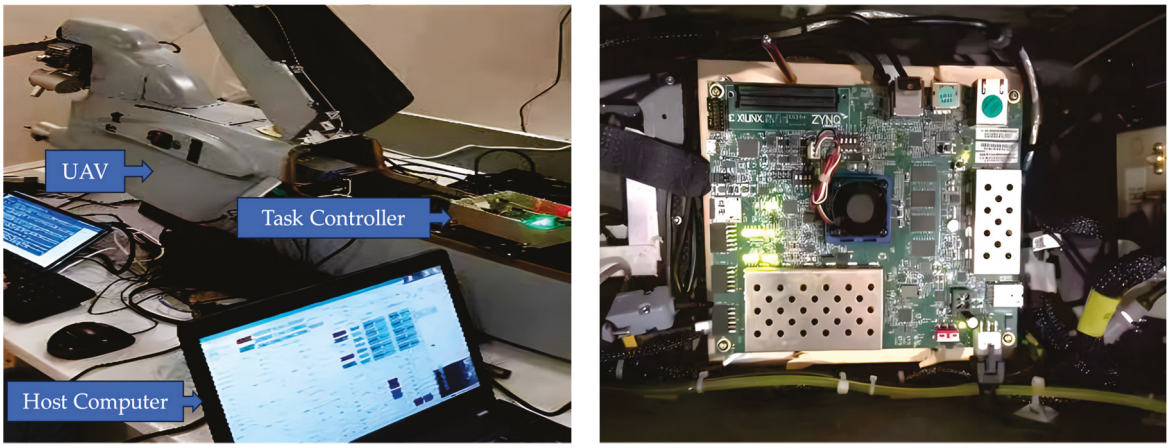


Figure 11. Schematic of the L-30A UAV platform and its components: (**a**) L-30A UAV system; (**b**) sensors and safety mechanisms.

4.1.2. Controller Selection and Algorithm Integration

(1) Controller selection and rationale

The ZCU104 platform, which combines an ARM Cortex-A53 processor and FPGA, was selected for its ability to handle intensive tasks like real-time trajectory planning and neural network inference, making it particularly suitable for multi-UAV control [38]. The FPGA accelerates computational tasks and provides the real-time processing capabilities necessary for UAV formation control. Figure 12 illustrates the ZCU104 hardware platform and its integration into the UAV control system.



(**a**) (**b**)

Figure 12. Experimental hardware platform and task controller. (**a**) Hardware platform; (**b**) task controller.

(2) Hardware performance evaluation and optimization

To evaluate the performance of the ZCU104 platform under different configurations, this study analyzes both execution time and average energy consumption. Tables 8 and 9 show comparisons of execution time in the default training mode and comparisons across different platforms in terms of energy consumption and runtime. The purpose of this evaluation is to assess the performance of the ARM Cortex-A53 processor running independently, as well as the performance improvement when combined with FPGA, compared to that of high-performance CPU platforms.

Table 8. Execution time comparison across platforms in default training mode.

Hardware Platforms	Model	Runtime (ms)	Frequency (GHz)
ARM	ARM-A53	0.5550	1
ARM+FPGA	ZCU104	0.3550	1

Table 9. Training runtime comparison across platforms.

Hardware Platforms	Model	Runtime (ms)	Average Energy Consumption (W)
ARM+FPGA	ZCU104	0.78	15
ARM	ARM-A53	15.2	5
CPU	E5-2630v4	0.45	85

Table 8 presents the execution time of the ARM Cortex-A53 processor and the ARM+FPGA combination in default training mode. While the ARM Cortex-A53 processor can handle basic tasks on its own, its execution time is significantly longer than that of the ARM+FPGA combination. By integrating FPGA, the ZCU104 reduces execution time substantially and demonstrates greater efficiency in handling complex tasks.

In contrast to the execution time comparison, Table 9 provides insights into the runtime and energy consumption of different hardware platforms, evaluated using the widely

recognized CoreMark benchmark. This benchmark, particularly well-suited for ARM architectures, highlights the significant differences in energy efficiency among platforms. While the Intel E5-2630v4 CPU delivers the fastest runtime, it comes at the cost of considerably higher energy consumption. On the other hand, the ZCU104 platform, which integrates both the ARM Cortex-A53 processor and FPGA, not only maintains processing efficiency but also drastically reduces energy consumption, making it an ideal choice for UAV control tasks where energy efficiency is paramount.

From the comparison in Table 9, it is clear that the ZCU104 platform, when combined with FPGA, offers a high energy efficiency ratio. Not only does it reduce the runtime for real-time tasks, but it also significantly lowers energy consumption. In contrast, while the Intel E5-2630v4 CPU demonstrates a faster runtime, its energy consumption reaches 85 watts, and its larger size and weight restrict its use as an onboard platform. Due to these limitations, the Intel E5-2630v4 is commonly used for high-performance ground simulations, but it is not suitable for onboard hardware applications.

In conclusion, the ARM+FPGA combination of the ZCU104 platform provides an optimal balance of performance and power efficiency, making it the best choice for UAV cluster control and flight validation tasks. Its efficient parallel processing capability and low energy consumption ensure reliability and stability for long-duration flight missions, making it particularly suitable for use as an onboard hardware platform in real-time flight validation.

4.2. Flight Validation and Performance Analysis

The validation of the MADDPG–VSC algorithm's performance is demonstrated through real-world formation flight experiments. The validation process emphasizes the algorithm's effectiveness in achieving cooperative control, obstacle avoidance, and overall UAV coordination. Key performance metrics, including trajectory tracking, formation stability, obstacle avoidance, and real-time responsiveness, are analyzed and discussed.

4.2.1. Experiment Process Overview

The flight experiment was conducted in a mission field measuring 2000 m $\times$ 1000 m $\times$ 1000 m, as shown in Figure 13, with the aim of validating the performance of the MADDPG–VSC and MADDPG algorithms on multi-UAV formation control tasks. The UAV flight trajectories were planned according to the ground truth coordinates, as shown in Table 10. The total flight duration was set to 535 s, with UAVs taking off and landing at the end of the runway. The initial position error was 0 m, and the UAVs maintained an average cruising speed of 20 m/s during the flight. To simulate more complex combat scenarios, the mission field was equipped with three radar detection zones representing enemy anti-aircraft artillery or ground fire threats. These threat zones provided a realistic operational environment for the UAV formation and further evaluated the algorithms' stability and adaptability in challenging conditions.

Table 10. Coordinates of ground truth waypoints.

Waypoint	**wp1**	**wp2**	**wp3**	**wp4**	**wp5**	**wp6**	**wp7**	**wp8**
Lon/° E	90.1437	90.1581	90.1767	90.1823	90.1618	90.1643	90.1520	90.1431
Lat/° N	38.3884	38.3939	38.4016	38.3901	38.3838	38.3763	38.3721	38.3882

The entire experiment was divided into four phases, i.e., takeoff, formation maintenance, mission execution, and safe landing. During the takeoff phase, the three UAVs simultaneously took off from designated starting points and maintained a triangular formation. In the mission execution phase, the UAVs flew along the predefined flight trajectories, adjusting the flight parameters based on real-time sensor data to ensure formation stability and minimize trajectory deviations. After completing the mission, the UAVs safely landed at the designated locations, marking the end of the flight.

To ensure the safety and reliability of the experiment, a series of comprehensive safety protection measures were implemented. The GCS closely monitored the UAV flight trajectories in real time using a feedback system, detecting any speed anomalies or heading deviations. Immediate adjustments or automatic return mechanisms were triggered, if necessary. In addition, a redundant communication system was employed to prevent signal loss or interference, ensuring stable data transmission between the UAVs and the GCS. During the flight, the UAVs were also equipped with parachutes and airbags to prevent collisions with other UAVs or ground obstacles. If any threats or disturbances occurred, the system would automatically deploy the parachutes, ensuring a safe landing.

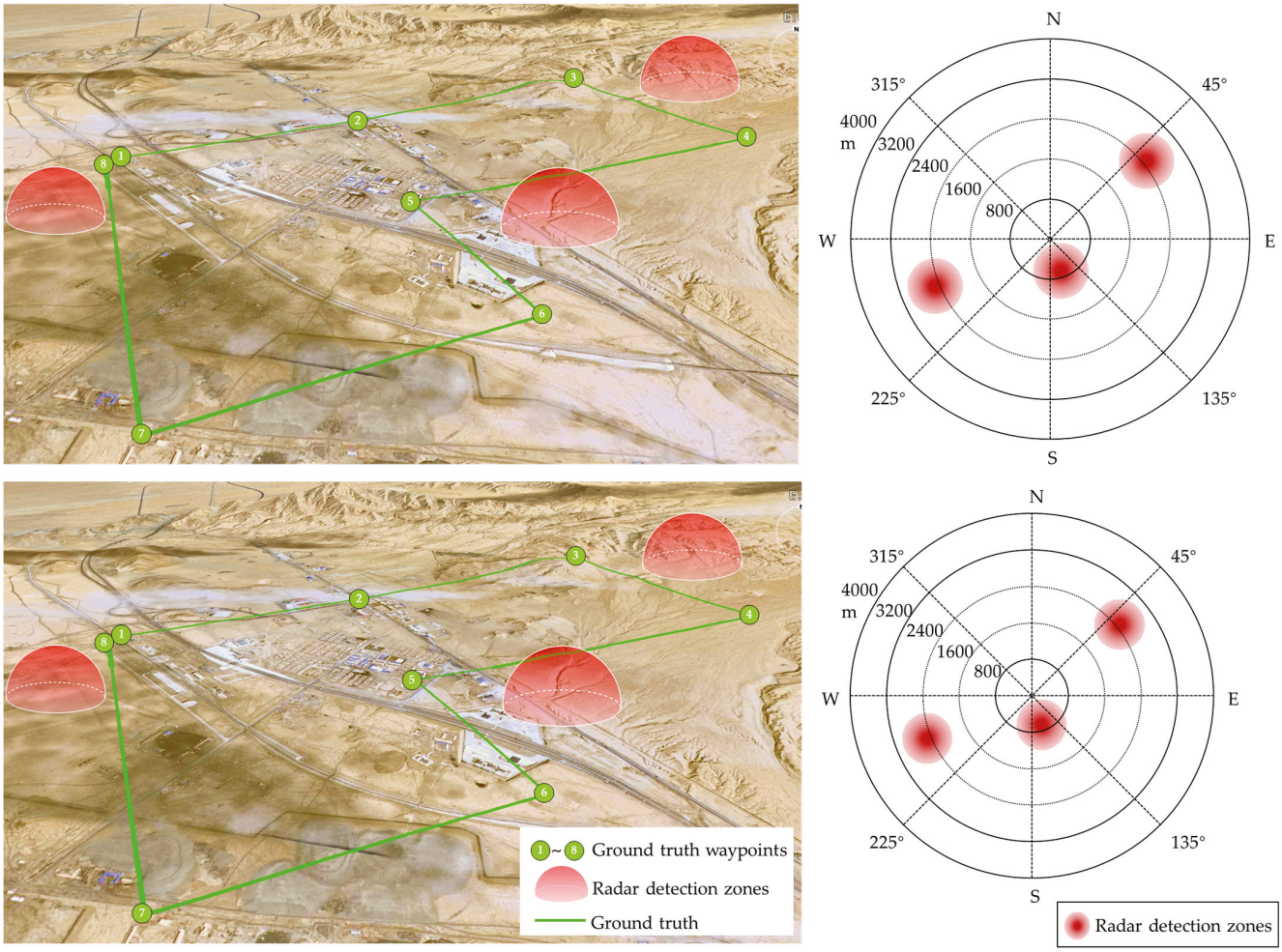


Figure 13. Scenario map of the formation flight trajectories.

4.2.2. Hardware Platform Ground Testing and Analysis

The evaluation of three hardware platforms, ZCU104, ARM-A53, and E5-2630v4, was conducted based on four key performance metrics, i.e., latency, energy consumption, throughput, and fault tolerance and reliability. These tests aim to determine the most suitable platform for multi-UAV coordination and control tasks, focusing on system performance under varying task complexities and real-time operational demands.

(1) Latency test

The latency test evaluates how quickly each hardware platform can respond to task commands by simulating real-time control scenarios in multi-UAV flight tasks, such as formation adjustments and obstacle avoidance. The test measures the response performance of each platform in both default training and training modes. To ensure accuracy, the tests

were conducted in a controlled local network environment, eliminating external network interference. The evaluation focuses on the response speed of each platform in single-task environments, particularly for tasks requiring high real-time performance.

As UAV formations grow, computational demands increase, which may affect real-time responsiveness. Optimization techniques like parallel processing and computational pruning are being explored to maintain low latency and improve scalability.

As shown in Figure 14, the E5-2630v4 platform exhibits relatively low latency, making it highly responsive to task commands. However, the ZCU104 platform, while displaying a longer overall runtime due to its FPGA's parallel processing, still performs competitively. In contrast, the ARM-A53 platform demonstrates significantly higher latency, particularly during intensive tasks, rendering it less suitable for applications that require fast, real-time responses.

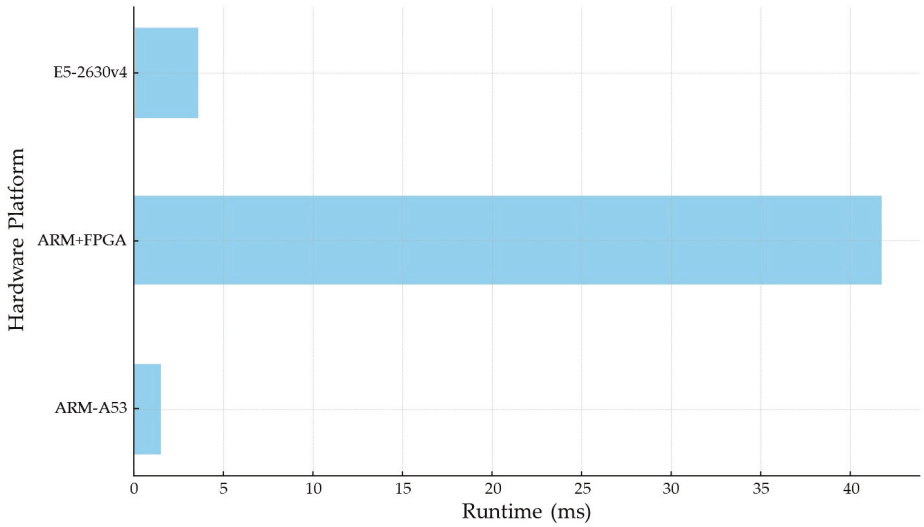


Figure 14. Latency comparison across different hardware platforms.

(2) Energy consumption test

The energy consumption test monitors the energy usage of each platform under multi-task loads, assessing performance across different task complexities. Tasks include cruising, formation adjustments, and obstacle avoidance, simulating typical multi-UAV operations. This test focuses on measuring energy consumption during task execution, particularly under high-load conditions.

Figure 15 shows the energy consumption of the three platforms. The ZCU104 platform demonstrates balanced energy consumption, handling complex tasks while maintaining a relatively low energy profile. The ARM-A53, with lower processing power, exhibits the lowest overall energy consumption, but its energy usage increases with task complexity. The E5-2630v4 platform exhibits significantly higher energy consumption due to its server-grade capabilities, making it less suitable for environments where energy efficiency is critical.

To optimize energy consumption in complex tasks, adaptive resource allocation and sleep modes are being explored, allowing the system to dynamically adjust processing power and prolong flight duration without sacrificing performance.

(3) Throughput test

The throughput test evaluates the capacity of each platform to handle multiple UAV tasks simultaneously. Multiple tasks were loaded to test the efficiency of each platform in multi-task environments. The goal of the throughput test is to measure the ability of each platform to process high loads of tasks, especially in scenarios requiring simultaneous control and data transmission.

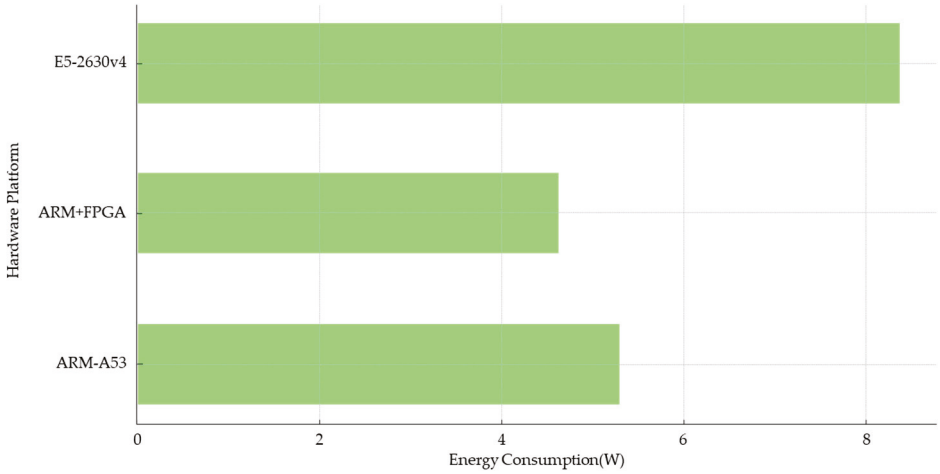


Figure 15. Energy consumption comparison across hardware platforms.

Table 11 shows that the E5-2630v4 platform performs best in terms of throughput, processing a large number of parallel tasks, making it ideal for high-density workloads. Although ZCU104's throughput is lower than E5-2630v4's, it remains stable and capable of supporting multi-UAV coordination tasks. The ARM-A53 platform shows the lowest throughput, struggling to handle multiple tasks efficiently, especially in complex multi-UAV operations.

Table 11. Throughput comparison of hardware platforms in default training mode.

Hardware Platforms	Model	Throughput (Tasks/s)
ARM	ARM-A53	100
ARM+FPGA	ZCU104	150
CPU	E5-2630v4	200

Real-time processing demands increase with larger UAV formations. Optimizations in task parallelization and data handling are being investigated to ensure that the hardware can manage higher task loads without sacrificing responsiveness.

(4) Fault tolerance and reliability

In complex flight environments, these two aspects are essential for maintaining system stability, as they assess different dimensions of platform performance. Fault tolerance focuses on how each platform manages system faults, communication errors, and unexpected failures during multi-UAV tasks, ensuring continuous operation with minimal disruption. Reliability, by contrast, evaluates the platform's consistency in performing without failures over extended periods, particularly under challenging mission conditions.

As shown in Figure 16, the ZCU104 platform exhibits superior fault tolerance due to its combination of ARM processing and the FPGA's flexibility. This allows the system to handle unexpected faults with minimal disruption, contributing to more stable UAV operations in dynamic conditions. In terms of reliability, ZCU104 also performs well, maintaining consistent operation under continuous workloads. While the E5-2630v4 demonstrates strong fault tolerance, its reliance on server-grade processing limits its adaptability to sudden failures in real-time operations. The ARM-A53 platform, with its limited processing power, shows lower fault tolerance, which can lead to system instability under high-load or fault-prone conditions. Its reliability also suffers under extended operational scenarios, making it less suitable for complex, high-demand missions.

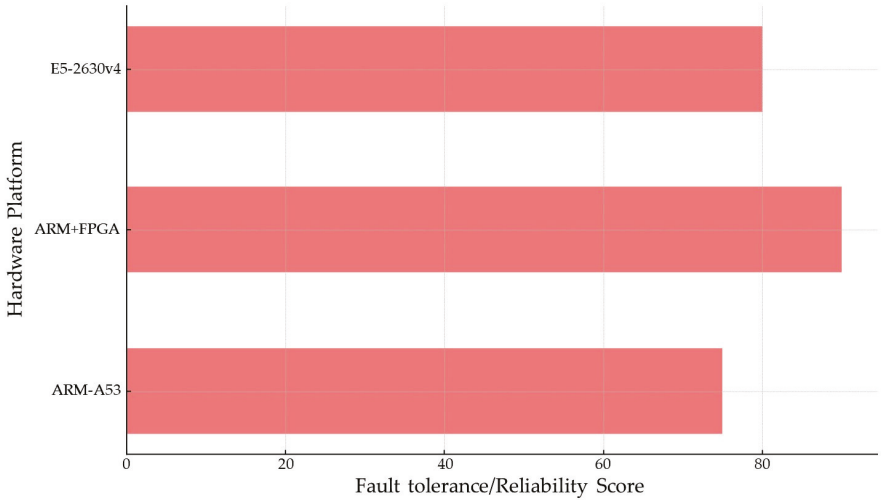


Figure 16. Fault tolerance and reliability comparison across hardware platforms.

Based on the evaluation of latency, energy consumption, throughput, and fault tolerance and reliability, the ZCU104 platform demonstrates a balanced performance across all metrics, making it the optimal choice for multi-UAV coordination and control tasks. Its competitive latency, efficient energy consumption, and stability in dynamic conditions make it well-suited for flight control missions. While the E5-2630v4 excels in computational power and throughput, its high energy consumption and relatively higher latency limit its suitability for real-time UAV control, rendering it more appropriate for ground-based simulations. The ARM-A53, despite its low energy consumption, lacks the necessary processing power, throughput, and fault tolerance to meet the demands of complex multi-UAV tasks, leading to its exclusion from further consideration.

4.2.3. Algorithm Performance and Data Analysis

This study provides a detailed analysis of the performance of the MADDPG–VSC and MADDPG algorithms in multi-UAV formation tasks, focusing on two key dimensions, i.e., coordination and real-time responsiveness. The analysis includes trajectory tracking error, formation stability, response time, and data transmission efficiency.

(1) Coordination evaluation

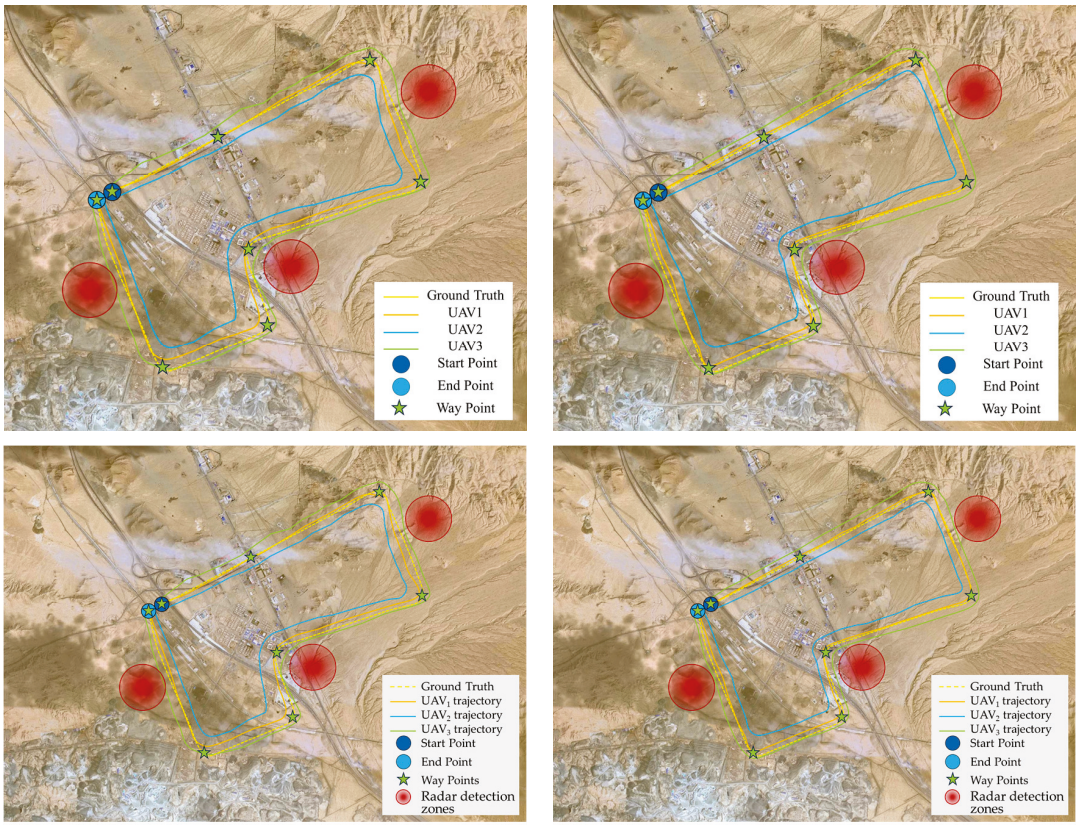
The evaluation of coordination focuses on two critical factors: trajectory tracking error and formation stability, which together measure the multi-UAV system's ability to maintain the formation and accurately follow the planned trajectory during mission execution.

- Trajectory tracking error

Figure 17 compares the trajectory tracking error of the center point of the UAV formation with the waypoints. The MADDPG–VSC algorithm demonstrated a smaller average error compared to that of the MADDPG algorithm. In scenarios with sharp turns and sudden changes, the MADDPG–VSC algorithm's adaptive control strategies helped reduce tracking errors, ensuring better accuracy.

- Formation stability

The stability of the UAV formation is measured by calculating the standard deviation of the relative distances between UAVs. Table 12 compares the formation stability of the MADDPG–VSC and MADDPG algorithms at various waypoints. The MADDPG–VSC algorithm consistently maintained lower standard deviations, indicating better stability, especially in complex mission scenarios.



(**a**) (**b**)

Figure 17. Trajectory tracking error comparison between MADDPG and MADDPG–VSC. (**a**) Trajectory of MADDPG; (**b**) trajectory of MADDPG–VSC.

Table 12. Formation stability comparison at different waypoints.

Waypoint	wp1	wp2	wp3	wp4	wp5	wp6	wp7	wp8
MADDPG	20.6 m	25.3 m	27.8 m	23.5 m	26.7 m	23.3 m	26.2 m	23.3 m
MADDPG–VSC	14.5 m	16.9 m	17.8 m	14.5 m	17.2 m	16.8 m	16.9 m	14.4 m

As shown in Table 12, the MADDPG–VSC algorithm maintains consistently lower standard deviations of relative distances across waypoints, indicating stronger formation stability compared to that of MADDPG.

(2) Real-time responsiveness evaluation

Response time, a critical indicator of the system's ability to quickly adjust, is measured as the time difference between when the GCS sends a command and receives feedback. As shown in Table 13, the MADDPG–VSC algorithm consistently outperformed the MADDPG algorithm, achieving an average response time of 1.2 s compared to 1.8 s for MADDPG, across various scenarios.

Table 13. Response time comparison between MADDPG–VSC and MADDPG in different scenarios.

Task Scenario	Avg Response Time (s)	
	MADDPG–VSC	MADDPG
Formation Adjustment	1.2	1.8
Obstacle Avoidance	1.3	2.0
Trajectory Recalibration	1.1	1.7

- Data transmission efficiency

The efficiency of data transmission is assessed by measuring the communication delay and transmission rate between the GCS and the UAVs. As shown in Table 14, the MADDPG–VSC algorithm demonstrated lower transmission delays, averaging 0.2 s with a transmission rate of 1.40 Mbps, compared to those of the MADDPG algorithm, which showed a delay of 0.3 s and a transmission rate of 1.30 Mbps.

Table 14. Data transmission delay and speed comparison between MADDPG–VSC and MADDPG.

Algorithm	Data Transmission Delay (s)	Data Transfer Rate (Mbps)
MADDPG–VSC	0.2	1.4
MADDPG	0.3	1.3

Through the analysis of coordination and real-time responsiveness, the MADDPG–VSC algorithm demonstrated significant advantages in multi-UAV formation tasks. It outperformed the MADDPG algorithm in terms of trajectory tracking error, formation stability, response time, and data transmission, particularly when handling complex and dynamic tasks where precise execution and stability are critical.

4.2.4. Key Insights and Implications

The validation experiments highlight the significant advantages of the MADDPG–VSC algorithm in multi-UAV formation tasks. Key performance indicators such as trajectory accuracy, formation stability, and real-time responsiveness demonstrate its ability to maintain precise control and coordination under dynamic conditions. The system's low latency and efficient data transmission further reinforce its effectiveness in real-time decision making.

These findings emphasize the innovations of the MADDPG–VSC algorithm, particularly in regards to adaptive sweep angle control and cooperation strategies. These enhancements provide a strong foundation for improving UAV performance in complex, real-world environments, ensuring reliable mission execution across a range of scenarios.

4.3. Comparative Analysis of Simulation and Flight Validation

(1) Trajectory and formation stability

The trajectory tracking accuracy and formation stability observed during real-world tests closely matched the simulated performance. Although some deviations were inevitable due to environmental factors, the average trajectory error and formation deviations remained within an acceptable range, underscoring the robustness of the control strategies implemented by the MADDPG–VSC algorithm.

(2) Hardware performance consistency

The hardware platforms, particularly the ZCU104 task controller, demonstrated stable performance across both simulation and real-world tests. The low latency, efficient data processing, and real-time control capabilities highlighted in the simulations were reflected in the flight tests, further validating the hardware selection.

(3) Response to innovation points

The adaptive sweep angle control and UAV cooperation mechanisms, which form the core innovations of the MADDPG–VSC algorithm, were shown to enhance the flexibility and stability of UAV formation in dynamic environments. While these features were rigorously tested in the simulations, the flight validation confirmed their effectiveness in real-world scenarios, particularly in maintaining precise control during sudden maneuvers or obstacle avoidance.

(4) Insights and implications for future work

The validation of the MADDPG–VSC algorithm through both simulation and flight testing provides confidence in its applicability for complex, real-world multi-UAV missions. Its ability to maintain formation stability, trajectory accuracy, and efficient obstacle avoidance in dynamic and unpredictable environments highlights its potential for broader implementation in UAV operations.

5. Conclusions

This study has made significant advancements in the modeling and control of variable-sweep wing UAVs, particularly through the refinement of longitudinal dynamic equations and the incorporation of maneuver-specific parameters from the L-30A model. These developments enabled accurate calculations of aerodynamic properties, such as optimal lift-to-drag ratios, and established a reliable model to represent the wing morphing process.

Additionally, the introduction of the MADDPG–VSC algorithm offers a novel approach for optimizing sweep angles in response to real-time flight states. This methodology enhances multi-UAV cooperative formation control, which is crucial for complex, real-world missions. Flight experiments and simulations show that the algorithm maintains precision, coordination, and stability in dynamic conditions, with minimal computational demands.

The MADDPG–VSC algorithm effectively addresses core challenges in UAV formation control, such as managing continuous action spaces and adapting to dynamic, unpredictable environments. By enhancing flight precision and formation stability, the algorithm shows strong potential for broader application in multi-UAV systems, particularly for tasks requiring high adaptability and scalability. This creates a robust framework for future use in increasingly complex flight environments.

In future work, real-world flight tests will be conducted under complex conditions, including varying weather and the presence of dynamic obstacles, to validate the algorithm's robustness in environments significantly different from the training data. An evaluation framework will be established to assess the impacts of unexpected changes, including sudden airflow shifts and target movements, on formation stability and control. In addition, future simulations will incorporate more complex and irregular obstacles to better reflect real-world challenges, such as sensor noise and communication delays. Efforts will also focus on improving the system's scalability and adaptability to support larger UAV formations and enhancing real-time decision making to further improve performance and operational reliability.

Author Contributions: Conceptualization, Z.C. and G.C.; methodology, G.C.; software, Z.C.; validation, Z.C. and G.C.; formal analysis, Z.C.; investigation, Z.C.; resources, Z.C.; data curation, Z.C.; writing—original draft preparation, Z.C.; writing—review and editing, G.C.; visualization, Z.C.; supervision, G.C.; project administration, G.C.; funding acquisition, G.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, Grant Nos. 92371201 and 52192633, and the Natural Science Foundation of Shaanxi Province, Grant No. 2022JC-03.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: Zhengyang Cao worked for Xi'an ASN UAV Technology Co., Ltd. Gang Chen declares no conflict of interest.

References

1. Botez, R.M.; Dao, T.M.; Elelwi, M.; Kuitche, M.A. Comparison and analyses of a variable span-morphing of the tapered wing with a varying sweep angle. *Aeronaut. J.* **2020**, *124*, 1146–1169. [CrossRef]
2. Hart, P.E.; Nilsson, N.J.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Man Cybern. Syst.* **1968**, *4*, 100–107. [CrossRef]
3. LaValle, S.M. *Planning Algorithms*; Cambridge University Press: Cambridge, UK, 2006; pp. 23–62.
4. Kavraki, L.E.; Svestka, P.; Latombe, J.C.; Overmars, M.H. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot. Autom.* **1996**, *12*, 566–580. [CrossRef]
5. Shadeed, O.; Hasanzade, M.; Koyuncu, E. Deep Reinforcement Learning based Aggressive Flight Trajectory Tracker. In Proceedings of the AIAA SciTech Forum 2021, Virtual Online, 11–15 January 2021. 1 PartF.
6. Bolinches, M.; Forrester, A.I.J.; Keane, A.J.; Scanlan, J.P.; Takeda, K. Design, analysis and experimental validation of a morphing UAV wing. *Aeronaut. J.* **2011**, *115*, 761–765. [CrossRef]
7. Valasek, J.; Tandale, M.D.; Rong, J. A Reinforcement Learning—Adaptive Control Architecture for Morphing. *J. Aerosp. Comput. Inf. Commun.* **2005**, *2*, 174–195. [CrossRef]
8. Lampton, A.; Niksch, A.; Valasek, J. Reinforcement Learning of a Morphing Airfoil-Policy and Discrete Learning Analysis. *J. Aerosp. Comput. Inf. Commun.* **2010**, *7*, 241–260. [CrossRef]
9. Lampton, A.; Niksch, A.; Valasek, J. Reinforcement Learning of Morphing Airfoils with Aerodynamic and Structural Effects. *J. Aerosp. Comput. Inf. Commun.* **2009**, *6*, 30–50. [CrossRef]
10. Valasek, J.; Doebbler, J.; Tandale, M.D.; Meade, A.J. Improved Adaptive–Reinforcement Learning Control for Morphing Unmanned Air Vehicles. *IEEE Trans. Syst. Man Cybern. B* **2008**, *38*, 1014–1020. [CrossRef] [PubMed]
11. Xu, D.; Hui, Z.; Liu, Y.; Chen, G. Morphing control of a new bionic morphing UAV with deep reinforcement learning. *Aerosp. Sci. Technol.* **2019**, *92*, 232–243. [CrossRef]
12. Yan, B.; Li, Y.; Dai, P.; Liu, S. Aerodynamic Analysis, Dynamic Modeling, and Control of a Morphing Aircraft. *J. Aerosp. Eng.* **2019**, *32*, 04019058. [CrossRef]
13. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
14. Silver, D.; Huang, A.; Maddison, C.J.; Guez, A.; Sifre, L.; van den Driessche, G.; Hassabis, D. Mastering the game of Go with deep neural networks and tree search. *Nature* **2016**, *529*, 484–489. [CrossRef] [PubMed]
15. Liu, F.; Lan, T.; Wang, S. A collaborative control method for multi-UAVs based on ADRC control. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020; pp. 7014–7019.
16. Ouamri, M.A.; Barb, G.; Singh, D.; Adam, A.B.M.; Muthanna, M.S.A.; Li, X. Nonlinear Energy-Harvesting for D2D Networks Underlaying UAV with SWIPT Using MADQN. *IEEE Commun. Lett.* **2023**, *27*, 1804–1808. [CrossRef]
17. Ji, H.; Tong, L. Multi-body dynamic modelling and flight control for an asymmetric variable sweep morphing UAV. *Aeronaut. J.* **2014**, *118*, 683–706. [CrossRef]
18. Rashid, T.; Samvelyan, M.; Witt, C.; Farquhar, G.; Foerster, J.; Whiteson, S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 4295–4304.
19. Liu, X.; Yin, Y.; Su, Y.; Ming, R. A Multi-UCAV Cooperative Decision-Making Method Based on an MAPPO Algorithm for Beyond-Visual-Range Air Combat. *Aerospace* **2022**, *9*, 56. [CrossRef]
20. Isci, H.; Koyuncu, E. Reinforcement Learning Based Autonomous Air Combat with Energy Budgets. In Proceedings of the AIAA SciTech Forum 2022, San Diego, CA, USA; Online, 3–7 January 2022. Autonomy I.
21. Du, Y.; Li, F.; Zandi, H.; Xue, Y. Approximating Nash Equilibrium in Day-ahead Electricity Market Bidding with Multi-agent Deep Reinforcement Learning. *J. Mod. Power Syst. Clean Energy* **2021**, *9*, 534–544. [CrossRef]
22. Koch, W.; Mancuso, R.; West, R.; Bestavros, A. Reinforcement Learning for UAV Attitude Control. *ACM Trans. Cyber-Phys. Syst.* **2019**, *3*, 1–21. [CrossRef]
23. Zhen, Z.; Zhu, P.; Xue, Y.; Ji, Y. Distributed intelligent self-organized mission planning of multi-UAV for dynamic targets cooperative search-attack. *Chin. J. Aeronaut.* **2019**, *32*, 2706–2716. [CrossRef]
24. Wang, C.; Wu, L.; Yan, C.; Wang, Z.; Long, H.; Yu, C. Coactive design of explainable agent-based task planning and deep reinforcement learning for human-UAVs teamwork. *Chin. J. Aeronaut.* **2020**, *33*, 2930–2945. [CrossRef]
25. Bekar, C.; Yuksek, B.; Inalhan, G. High Fidelity Progressive Reinforcement Learning for Agile Maneuvering UAVs. In Proceedings of the AIAA SciTech Forum 2020, Orlando, FL, USA, 6–10 January 2020. 2020-0898.
26. Elkins, J.G.; Sood, R.; Rumpf, C. Bridging Reinforcement Learning and Online Learning for Spacecraft Attitude Control. *J. Aerosp. Inf. Syst.* **2021**, *19*, 62–69. [CrossRef]
27. Duryea, E.; Ganger, M.; Hu, W. Exploring Deep Reinforcement Learning with Multi Q-Learning. *Intell. Control Autom.* **2016**, *7*, 129–144. [CrossRef]
28. Zhang, Y.; Wu, Z.; Ma, Y.; Sun, R.; Xu, Z. Research on autonomous formation of Multi-UAV based on MADDPG algorithm. In Proceedings of the 2022 IEEE 17th International Conference on Control & Automation (ICCA), Naples, Italy, 27–30 June 2022; pp. 249–254.

29. Bouhamed, O.; Ghazzai, H.; Besbes, H.; Massoud, Y. Autonomous UAV Navigation: A DDPG-Based Deep Reinforcement Learning Approach. In Proceedings of the 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 12–14 October 2020; pp. 1–5.
30. Zhou, W.; Li, J.; Liu, Z.; Shen, L. Improving multi-target cooperative tracking guidance for UAV swarms using multi-agent reinforcement learning. *Chin. J. Aeronaut.* **2022**, *35*, 100–112. [CrossRef]
31. Chen, Y.J.; Chang, D.K.; Zhang, C. Autonomous Tracking Using a Swarm of UAVs: A Constrained Multi-Agent Reinforcement Learning Approach. *IEEE Trans. Veh. Technol.* **2020**, *69*, 13702–13717. [CrossRef]
32. Yan, B.; Dai, P.; Liu, R.; Xing, M.; Liu, S. Adaptive super-twisting sliding mode control of variable sweep morphing aircraft. *Aerosp. Sci. Technol.* **2019**, *92*, 198–210. [CrossRef]
33. Hernandez-Leal, P.; Kartal, B.; Taylor, M.E. A survey and critique of multiagent deep reinforcement learning. *Auton. Agents Multi-Agent Syst.* **2019**, *33*, 750–797. [CrossRef]
34. Gaudet, B.; Linares, R.; Furfaro, R. Six Degree-of-Freedom Hovering over an Asteroid with Unknown Environmental Dynamics via Reinforcement Learning. In Proceedings of the AIAA SciTech Forum 2020, Orlando, FL, USA, 6–10 January 2020. 2020-0953.
35. Kashyap, V.; Vepa, R. Reinforcement learning based Linear quadratic Regulator for the Control of a Quadcopter. In Proceedings of the AIAA SciTech Forum 2023, National Harbor, MD, USA; Online, 23–27 January 2023. 2023-0014.
36. Webb, K.; Rogers, J. Adaptive Control Design for Multi-UAV Cooperative Lift Systems. *J. Aircr.* **2021**, *58*, 1302–1322. [CrossRef]
37. Dai, J.; Wang, S.; Jang, Y.; Wu, X.; Cao, Z. Multi-UAV cooperative formation flight control based on APF & SMC. In Proceedings of the 2017 2nd International Conference on Robotics and Automation Engineering (ICRAE), Shanghai, China, 29–31 December 2017; pp. 222–228.
38. Zynq UltraScale+ MPSoC ZCU104 Evaluation Kit. Available online: https://docs.xilinx.com/v/u/en-US/ug1267-zcu104-eval-bd (accessed on 9 October 2018).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

Sequence-Information Recognition Method Based on Integrated mDTW

Boliang Sun and Chao Chen *

Laboratory for Big Data and Decision, National University of Defense Technology, Changsha 410073, China; sunboliang1986@126.com

* Correspondence: chenc1997@nudt.edu.cn

Abstract: In the fields of machine learning and artificial intelligence, the processing of time-series data has been a continuous concern and a significant algorithm for intelligent applications. Traditional deep-learning-based methods seem to have reached performance ceilings in certain specific areas, such as online character recognition. This paper proposes an algorithmic framework to break this deadlock by classifying time-series data by evaluating the similarities among handwriting samples using multidimensional Dynamic Time Warping (mDTW) distances. A simplified hierarchical clustering algorithm is employed as a classifier for character recognition. Moreover, this work achieves joint modeling with current mainstream temporal models, enabling the mDTW model to integrate modeling results from methods like RNN or Transformer, therefore further enhancing the accuracy of related algorithms. A series of experiments were conducted on a public database, and the results indicate that our method overcomes the bottleneck of current deep-learning-based methods in the field of online handwriting character recognition. More importantly, compared to deep -learning-based methods, the proposed method has a simpler structure and higher interpretability. Experimental results demonstrate that our proposed method outperforms existing state-of-the-art models in handwriting character recognition, achieving a top-1 accuracy of 98.5% and a top-3 accuracy of 99.3%, thus confirming its effectiveness in overcoming the limitations of traditional deep-learning models in temporal sequence processing.

Keywords: computational intelligence; multidimensional dynamic time warping (mDTW); online character recognition; deep-learning model integration

1. Introduction

Measuring the similarities of different samples is fundamental to many fields of computational intelligence. Many algorithms take distances to value similarities of samples for classification, detection, and other pattern-recognition tasks. In most of the research, samples are represented by a set of vectors through encoding processes. Then, the Minkowski distances of those vectors could be computed as similarities. Minkowski distances are defined as $dist(x, y) = \sum_{k=1}^{d} (x_k - y_k)^p$, which is one of the most popular metrics to value samples distances. When $p = 1$, Minkowski distance is named Manhattan distance; when $p = 2$ it is called Euclidean distance. Another popular measure is Mahalanobis distance, which could be viewed as distorted Euclidean distance. Overall, these studies highlight the need for most of the current research.

In the past decade, significant progress has been made in sequence-information processing. Sequence-information processing refers to the analysis and manipulation of data sequences, such as text, speech, or biological sequences. This field involves techniques like natural language processing (NLP) and sequence alignment, which enable machines to understand, generate, and transform sequential data efficiently. However, there is still room for improvement in related methods, especially in terms of interpretability, model integration, and other aspects. Existing methods do not perform well in explaining the

complex temporal structure inherent in the data [1,2]. As a remedy, Dynamic Time Warping (DTW) [3] has emerged as a compelling alternative [4–6]. DTW's most notable advantage lies in its resilience against signal warping, encompassing shifts and scaling along the time axis or even the Doppler effect. Consequently, DTW has evolved into one of the most favored metrics for tasks involving pattern matching. For example, when dealing with signals from different sampling frequencies, the conventional point-wise Euclidean distance may yield different results, especially when one signal is merely a compressed version of another. Without using mDTW for sequence point registration, it is not possible to determine which two points are directly used to calculate the Euclidean distance. Therefore, for two time series, the use of DTW is necessary. In this case, DTW performs well by expertly capturing the scale variation and the minimum distance between the generated signals. What sets DTW apart is not only its ability to produce distance values but also its ability to reveal the exact alignment of two sequences. DTW is unique in that it not only computes distance values but also aligns two sequences precisely, which is particularly critical in intelligent computing applications. Moreover, DTW is not just a similarity measure. As a robust feature-extraction tool, it enhances its versatility and value in the context of intelligent computing.

The processing of online character data is a typical and formidable challenge in the field of time-series processing. Accurate and efficient recognition of handwritten characters is of great significance, especially in an environment where smart devices are constantly evolving. Over recent decades, the domain of sequence-information processing, including non-rigid sparse point matching, Handwritten Chinese Character Recognition (HCCR), and Human Action Recognition (HAR), has become a research hotspot and has made remarkable achievements and progress. Furthermore, the intricacies embedded in the Chinese written language, characterized by a vast number of character classes, introduce an additional layer of complexity to the recognition task. The sheer volume of characters, each with its unique features, poses a formidable challenge for researchers and practitioners alike. The intricate structure inherent in most Chinese characters further amplifies the difficulty of the HCCR task, contributing to its status as a persistent and demanding challenge within the research community.

The proposed model presented in this paper offers high interpretability, integrating traditional sequence modeling methods with deep-learning techniques to achieve superior performance. Its transparent framework ensures a clear understanding of each component and parameter, exemplified by the straightforward calculation process of mDTW. Furthermore, the model's seamless integration with deep learning enables powerful feature extraction while maintaining interpretability. With low computational complexity and minimal parameterization, the model remains efficient and reliable, even for large-scale datasets. Overall, this approach combines interpretability, deep-learning integration, and computational efficiency, making it well-suited for a wide range of intelligent computing tasks.

Our Contribution

Development of the mDTW Algorithm: We developed a multidimensional Dynamic Time Warping (mDTW) algorithm specifically designed for handling sequence information, significantly enhancing accuracy in tasks such as online handwriting recognition.

Integration with Deep-Learning Models: We proposed the integration of mDTW with modern deep-learning models such as Recurrent Neural Networks (RNNs) and Transformers, which improved sequence-data classification across multiple domains, including text recognition, action classification, and point matching.

Hierarchical Clustering Model: We introduced a simplified hierarchical clustering model that groups characters and reduces the computational time for DTW distance calculations, making it feasible to efficiently process large handwriting datasets.

2. Related Works

2.1. Related Works of DTW

DTW (Dynamic Time Warping) [3] distance is a method employed to calculate the similarity between two time series. It was initially proposed and utilized in the field of speech recognition and has subsequently found wide application in various domains, including time-series classification, pattern recognition, anomaly detection, clustering analysis, and more. DTW excels at finding the best match between two sequences by elastically stretching or compressing the time series, enabling effective comparison and matching even when the sequences exhibit offsets or rate changes on the time axis. This is achieved by identifying the best alignment (i.e., the optimal path) between the two sequences, which minimizes the sum of the pairwise distances along this path, therefore accurately reflecting the similarity between the two sequences. This alignment allows for nonlinear stretching of the sequences on the time axis, making DTW particularly suitable for handling time series of different lengths or with varying rates.

In time-series classification tasks, DTW serves as a valuable similarity measure, aiding in the determination of what category a sequence belongs to. By calculating the DTW distance between the sequence to be classified and sequences of known categories, the category with the smallest distance can be confidently selected as the classification result. Furthermore, DTW can be utilized to identify specific patterns in time series or detect anomalies. By comparing a time series with a known pattern or a sequence within the normal range and calculating the DTW distance between them, it becomes possible to determine whether the time-series matches the expected pattern or deviates from the normal range.

In time-series clustering tasks, DTW distance proves to be an effective distance measure in clustering algorithms. By calculating the DTW distance between different sequences, similar time series can be clustered together, enabling the discovery of sequence groups with similar morphologies but possible offsets on the time axis.

DTW stands out as a potent and versatile tool within the realm of temporal data mining, finding applications across various domains due to its unique capabilities. One illustrative domain where DTW demonstrates its prowess is in acoustic data analysis, where it exhibits invariance to Doppler effects. This particular attribute makes DTW invaluable for tasks involving the precise alignment and comparison of acoustic signals, regardless of potential frequency shifts induced by motion or other factors.

Expanding the scope of DTW's applications, it plays a crucial role in the analysis of biological signals such as Electrocardiogram (ECG) [7] or Electroencephalogram (EEG) [8,9]. Researchers leverage DTW to effectively characterize and discern patterns within these signals, aiding in the identification and understanding of potential diseases. The flexibility of DTW in handling diverse types of temporal data makes it an indispensable tool for extracting meaningful insights from complex biological signals.

Moreover, in the context of time-series classification problems, DTW serves as a feature extractor when combined with predefined patterns or features, as highlighted in the work by Kate et al. [10]. This approach allows DTW to contribute significantly to the classification of time-series data, where the alignment of sequences is essential for accurate pattern recognition. The use of Hamming distance [1,11] in conjunction with DTW alignment in this setting is referred to as Edit distance, a well-studied variant that enhances our understanding of the relationships between different temporal sequences [12].

In essence, the versatility of DTW spans diverse applications, from mitigating Doppler effects in acoustic data to aiding in the diagnosis of diseases through the analysis of biological signals. Its role as a feature extractor in time-series classification further underscores its adaptability and utility in extracting meaningful information from temporal datasets. The continued exploration and application of DTW in various domains highlight its enduring significance in the field of temporal data mining and analysis.

The proposed framework presented in this paper is combined with two parts: a multi-dimension DTW (mDTW) [13] algorithm and a hierarchical cluster model. The core

idea of our work is very simple: compute similarities among online handwriting samples and use them to classify characters. However, a major problem with the experimental method is that the training time cost is too long. In particular, when we are dealing with a handwriting database that has N_c characters and each character written by N_w writers, the DTW process should be repeated $(N_c \times N_w)^2/2$ times to establish a cluster model. Since a Chinese handwriting database often has thousands of characters and hundreds of writers, computing the DTW so many times would take almost a decade on most computing devices, which is not acceptable for researchers. Therefore, we put forward the above two methods to decrease time costs.

2.2. Related Works of Sequence Data

Time-series classification and regression are important techniques for analyzing and modeling data that changes over time. These methods have wide-ranging applications, including Human Activity Recognition, Handwritten Chinese Character Recognition, Earth Observation, Medical Diagnosis, and more. In this study, we will focus on the application of time-series classification and regression in Human Activity Recognition and Handwritten Chinese Character Recognition and provide an overview of the latest developments and challenges in these fields.

Human activity recognition (HAR) is the identification or monitoring of human activity through the analysis of data collected by sensors or other instruments [14]. The recent growth of wearable technologies and the Internet of Things has resulted not only in the collection of large volumes of activity data [15] but also in the easy deployment of applications utilizing these data to improve the safety and quality of human life. HAR is, therefore, an important field of research with applications including healthcare, fitness monitoring, smart homes [16], and assisted living [17].

Handwritten Chinese Character Recognition (HCCR) is a field of study focused on the automatic recognition of handwritten Chinese characters. This technology plays a crucial role in various applications, such as document digitization, optical character recognition (OCR) systems for forms, checks, and IDs, and handwriting input devices. Since the 1980s, HCCR has been a significant research area in pattern recognition due to the complexity and wide variety of Chinese characters. For instance, in Liu et al.'s paper [18], discriminative feature-extraction methods and discriminative learning quadratic discriminant function classifiers were employed, achieving outstanding recognition rates on challenging online and offline HCCR datasets CASIA-OLHWDB and CASIA-HWDB. The best online recognition rates for single characters were 95.28% (DB1.0, 4037 characters), 94.85% (DB1.1, 3926 characters), and 95.31% (ICDAR 2013 Competition DB, 3755 characters). The best offline recognition rates were 94.20% (DB1.0), 92.08% (DB1.1), and 92.72% (ICDAR 2013 Competition DB). More recently, Ren et al. [19]. utilized an RNN, combining Bidirectional Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU), which processes raw sequential data directly without converting online handwriting trajectories into image-like representations, achieving an end-to-end recognition rate of 98.15% on the ICDAR-2013 dataset.

Compared to other related works, it has been challenging to achieve better performance through deep-learning training. Collectively, there seems to be some evidence to indicate that existing Deep-learning-based HCCR methods have reached the performance bottleneck. Zhang et al. [20] achieved groundbreaking results in the field of HCCR by integrating traditional direction-decomposed feature maps with deep convolutional neural networks; however, it is important to note that these achievements, while groundbreaking, are now seven years old. Additionally, Li et al. [21] introduced a novel approach in the HCCR domain leveraging a Siamese neural network to predict the similarity between handwritten Chinese characters and template images, showing promising generalization to new classes not seen in training, but the method may struggle with limited data scenarios and the complexity of optimizing Siamese networks. However, by integrating mDTW and

Transformer, a significant performance breakthrough was successfully achieved with this framework, demonstrating its potential in other specific sequence-data-related domains.

3. Methodology

As mentioned earlier, the mDTW method presented in this paper serves as a preliminary processing step for classifying data and selecting the top-N results (the top N most likely classifications) to form a rough classification probability distribution. Subsequently, this rough distribution undergoes further refinement through techniques such as cGRU to achieve a more accurate classification. As shown in Figure 1.

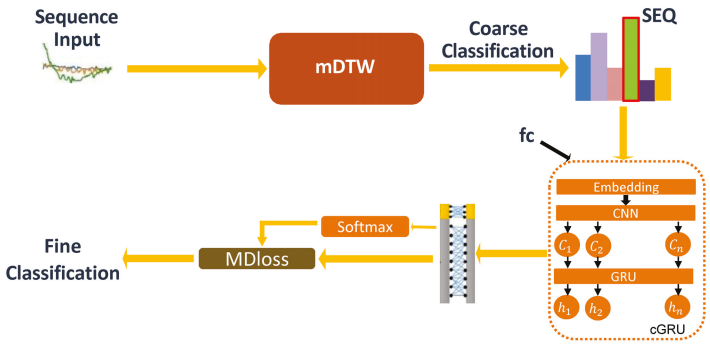


Figure 1. Schematic diagram of the core framework process.

3.1. Multidimensional DTW

Similar to the conventional one-dimensional Dynamic Time Warping (DTW) approach, the method presented here encompasses two primary processes: the forward process and the backward process. In the fundamental stage, the algorithm computes the point-to-point distance matrix, a critical step in establishing the foundational structure for subsequent analyses. This matrix serves as the basis for understanding the dissimilarity between corresponding points in the sequences. Following this, the warping process comes into play, where the optimized point-matching relationships are computed.

In the forward process, the algorithm systematically evaluates the distances between each point in the sequences, creating a comprehensive matrix that encapsulates the pairwise dissimilarities. This matrix lays the groundwork for subsequent alignment, forming the foundation upon which the algorithm can identify the optimal alignment path. The backward process is an equally crucial component, focusing on the derivation of the optimal warping path based on the accumulated point-to-point distances. By iteratively backtracking through the matrix, the algorithm refines the matching relationship between points, ultimately arriving at an alignment that minimizes the overall dissimilarity. Together, these dual processes, the forward and backward stages, synergistically contribute to the effectiveness of the proposed method in capturing the nuanced relationships and achieving optimal alignment between two sequences using the Dynamic Time Warping technique.

Given a couple of sequences $\{SEQ_1 \in \mathcal{R}^{n\times d}, SEQ_2 \in \mathcal{R}^{m\times d}\}$, mDTW first computes a Dm matrix by Equation (1)

$$Dm_{ij} = dsum(\alpha, Seq_{1i}, Seq_{2j}) \tag{1}$$

The variables in formulas are as follows: α is a weight vector of different values in Seq, and $dsum$ is a user-defined formula. Data format of online Chinese handwriting sample is configured as $seq \in \mathcal{R}^{n\times 7}$, each temporal slice $seq_i = \{x_i, y_i, \Delta x_i, \Delta y_i, s_i\}$. Where n is the number of points in a handwriting sequence, $x_i, y_i \in \mathcal{R}$ are the location of a point, $\Delta x_i, \Delta y_i \in \mathcal{R}$ are pen tip displacement of the same point and $s_i \in \{0, 1\}$ is pen tip state. Here $s_i = \{1, 0, 0\}$ indicates the pen tip will continue the writing process, $s_i = \{0, 1, 0\}$ indicates

stroke ends and the pen tip will be raised and moved to another location, $s_i = \{0, 0, 1\}$ indicates the end of the character. Then $dsum$ is defined as Equations (2)–(5).

$$dsum = \alpha_1 \cdot ld + \alpha_2 \cdot dd + \alpha_3 \cdot sd \tag{2}$$

$$ld = \sqrt{(x_{1i} - x_{2j})^2 + (y_{1i} - y_{2j})^2} \tag{3}$$

$$dd = \sqrt{(\Delta x_{1i} - \Delta x_{2j})^2 + (\Delta y_{1i} - \Delta y_{2j})^2} \tag{4}$$

$$sd = \beta_1 |s_{i1} - s_{j1}| + \beta_2 |s_{i2} - s_{j2}| + \beta_3 |s_{i3} - s_{j3}| \tag{5}$$

ld, dd, sd are location distance, displacement distance, and state distance, respectively. ld represents the Euclidean distance between two points from different sequences, which directly reflects the spatial distance relationship between the relevant points. It is the most important constraint for our mDTW point matching. dd represents the Euclidean distance between the difference vectors of two points and their previous sampling time steps, which can be understood as the similarity in direction. Its calculation formula is $\Delta X = X_t - X_{t-1}$. For example, in a plane, if the difference vectors of two sequence points both indicate that the movement direction of the curve/trajectory at this time is to the left, then they should naturally obtain a certain degree of matching similarity. A higher value of α_i means we place more attention on the similarity of the corresponding factor to match the points. Since the information on location is contained in displacement data, much of the deep-learning-based handwriting research only relies on displacement. However, the cluster model in this paper does not have a parameter optimizing process to learn the information of location from displacement, so we added ld in Equation (3) to enhance the matching performance. β is the weight of writing states. Considering one character has almost 100 points and 5–10 strokes means that the rate of end stroke is about 10% and character end is 1%, β is set as $[1, 10, 100]$ to balance the infect of different writing states.

The dimension of a computed matrix Dm is $n \times m$, and each element represents $dsum$ distance of two points. The second step of mDTW is computing a matrix G for final point matching through an iterative process. This process is computed point-by-point by Equation (6)

$$G(i, j) = \min \begin{cases} G(i-1, j) + Dm(i, j) \\ G(i-1, j-1) + \lambda \cdot Dm(i, j) \\ G(i, j-1) + Dm(i, j) \end{cases} \tag{6}$$

λ is the weight of temporal similarity, which is configured as a constant 2 in traditional DTW algorithms. In this paper, we made it flexible to fit the multi-dim point matching task. G is computed from bottom to the top, form left to right. As shown in Figure 2, DTW has following steps in summery:

- Step 1: compute Dm Equations (2)–(5)
- Step 2: initialize G as a zero matrix
- Step 3: compute the first row (G_1) of G by Equation (6) form G_{11} to G_{1n} (green values in Figure 2)
- Step 4: repeat Step 3 m times until all elements in G are updated
- Step 5: start from G_{mn}, jump to a temporal location via $min\{G_{i,j-1}, G_{i-1,j-1}, G_{i-1,j}\}$.
- Step 6: Repeat Step5 until terminal G_{11} (red arrows in Figure 2)

Completing the above steps will return the matrix G, where G_{mn} could be recognized as the final distance of two sequences $DTWdist(seq_i, seq_j)$. Meanwhile, the computing process has a very high time cost in most of the corresponding tasks especially when dealing with complex sequence problems. Therefore, the algorithm is accelerated by parallel computing for practical application. The method of acceleration uses GPU to do high-dimension matrix multiplications. In the case of ensuring that the GPU memory limit is not exceeded, we could set a batch of sequences to do DTW computation parallel. This method further reduces the calculation time cost of the system.

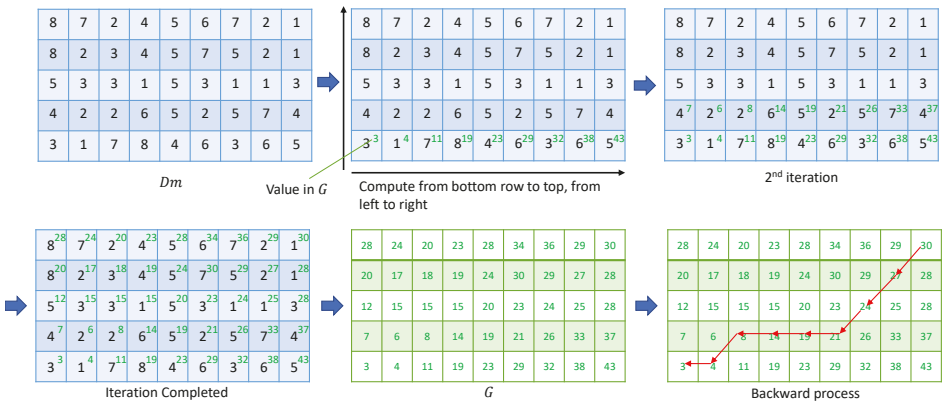


Figure 2. Process of dynamic warping, including forward and backward steps.

3.2. Sequence Clustering

The clustering process outlined in this paper can be interpreted as a form of supervised modeling. Specifically, the method introduced here does not involve the computation of distances between distinct characters; rather, it focuses solely on samples within the same character classes. As elucidated earlier, the computation of complete inner Dynamic Time Warping (DTW) distances for an extensive Chinese handwriting database demands a considerable amount of time, rendering it impractical for experimental purposes. Consequently, we propose the adoption of a simplified hierarchical clustering model designed to compute distances exclusively within the confines of the same character classes. Configuring the model in this manner not only addresses a specific challenge but also confers a distinct advantage, particularly in circumventing the potential computational overhead associated with unnecessary DTW calculations. This streamlined approach serves as a pivotal strategy in mitigating the time cost incurred during the training of the model. The careful restriction of computations to inner class distances within our proposed hierarchical clustering model is a deliberate choice that significantly optimizes the efficiency of the training process.

By narrowing the scope of calculation to inner class distances, we strategically eliminate the need for superfluous DTW calculations, leading to a more efficient and resource-friendly model training procedure. This deliberate simplification ensures that the model focuses its efforts on the most relevant aspects, contributing to a streamlined training phase. As a result, the proposed hierarchical clustering model not only expedites the training process but also alleviates the computational burden associated with analyzing the entire database. In essence, the methodology we propose is designed to strategically minimize the intricacies typically involved in DTW computations. The model, by concentrating exclusively on inner class distances, enhances both the expediency and effectiveness of the clustering process. This approach signifies a purposeful optimization, prioritizing efficiency without compromising the model's ability to discern meaningful patterns within the data. By adopting this strategic simplification, the proposed model emerges as a more efficient and scalable solution, making it well-suited for applications where computational resources are a critical consideration. In summary, the deliberate narrowing of focus within our hierarchical clustering model is a strategic choice that pays dividends in terms of both time efficiency and computational effectiveness, ultimately contributing to an enhanced clustering performance.

Here, we introduce an exclusive concept of this paper: useless DTW calculation. The definition of this is the computation of DTW distance between two Chinese characters with obvious differences in stroke structures. For instance, when we distinguish Chinese characters from dissimilar handwriting samples, the cluster centers could hardly effect testing results since the results (value of DTW) of these samples are too big to be recognized as the same class. Therefore, the computation of those DTW distances is useless. On the contrary, choices of cluster centers are very important for similar Chinese characters;

appropriate cluster centers could remarkably increase recognizing accuracy, as shown in Figure 3. Digits are DTW values between the testing center and samples. For characters with high dissimilarities (gray samples), the value of DTW is much higher than similar ones (blue samples). The value destabilization coursed by center changing could not affect the final results for gray samples but have a significant impact on similar samples.

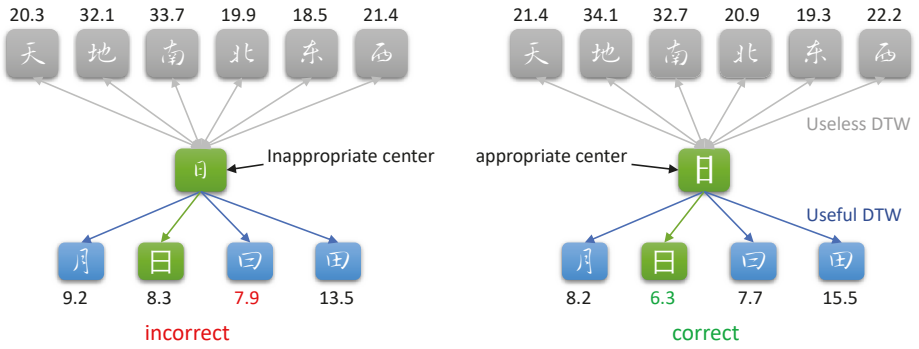


Figure 3. The definition of useless DTWs.

Therefore, the proposed method ignores computing useless DTWs on dissimilar characters and only considers similar samples instead. In order to further reduce the training time cost, the clustering process is set to compute DTWs between samples with the same characters. As discussed above, a database with $N_c \times N_w$ samples urges repeating DTW $(N_c \times N_w)^2/2$ times at the origin. The simplified method needs to repeat the DTW process $N_c \times N_w^2/2$ times only. For a handwriting database with thousands of characters ($N_c > 1000$), this step could reduce more than 99.9% of total origin training time cost.

Unlike the cluster methods based on Minkowski distances, a major problem within the sequence cluster domain is that it is very hard to generate a virtual sequence as a "center" in the corresponding space. Therefore, the proposed method should take some of the real samples as cluster centers by a hierarchical model. Then, the concrete steps for clustering a $N_c \times N_w$ database of this work are arranged as follows:

- Step 1 Separate the whole database into N_c groups by their characters. Thus, each group has N_w handwriting samples with the same character.
- Step 2 Choose a cluster number $n > 0 \in \mathcal{R}$ and then divide each group to cl_n clusters by minimizing Equation (7)

$$\arg\min[\sum_{i=1}^{n}\sum_{k=1}^{cl_i} DTWdist(Cen_i, Seq_k)] \tag{7}$$

  where $Center_i$ is the center of i_{th} cluster, cl_i is the number of sequence samples in i_{th} cluster, Seq_k is the k_{th} sequence sample of i_{th} cluster. The significance of this step is finding a set of centers that could minimize the sum of DTW distances between training samples and the nearest appropriate centers. In other words, this step finds n real samples as the representation of a whole sample group.
- step3 Repeat step2 N_c times to obtain all characters clustered. After that, we obtain a center set with $N_c \times n$ sequence samples.

Step 2 is the core content of the proposed cluster framework. To realize this, the following calculation steps were adopted, as listed in Algorithm 1. Where D is the storage of DTW distance for every two samples in the group, Equation (8) shows the computation of this process.

$$D_{ij} = DTWdist(seq_i, seq_j) \tag{8}$$

where seq_i represents the ith time-series vector. It should be noted that for temporal information, its vector may be discontinuous. For example, in handwriting, a word may be composed of multiple strokes. Therefore, when performing mDTW calculation on two time

series of text, we need to consider the matching relationships between different strokes. Noticing that D here is a symmetric matrix, and all the diagonal elements are 0. Then a temporal matrix D' is generated from D by Equation (9).

$$D_i' = D_i \times \frac{iter}{iter+1} + D_j \tag{9}$$

After the iteration process in line2–line6 of Algorithm 1, $C \in \mathcal{R}^{n \times N_w}$ is calculated. Each line of C shows the sequence index of a cluster. Then, we calculate the centers of every cluster. For instance, the non-zero elements in i_{th} row of C are $c_{i1}, \ldots, c_{im}$. Finally, find the center of this cluster by Equation (10).

$$Cen(k) = \arg\min\left(\sum_{l=1}^{m} DTWdist(Seq_k, Seq_{c_{il}})\right) \tag{10}$$

Algorithm 1: Hierarchical cluster for a Chinese handwriting sample group

Input: A group of training sequences $SEQ\{seq_1, \ldots seq_{N_c}\}$
0: Initialize a cluster index matrix $C \in \mathcal{R}^{N_c \times N_c}$, set $C_{i1} = i$ and others as 0
1: Compute a DTW distance matrix $D \in \mathcal{R}^{N_c \times N_c}$ via Equation (8).
2: for iter in $Nc - n$
3: find the location $i, j = argmin(D), (i < j)$
4: compute a temporal matrix D' by Equation (9) and replace D by D'
5: merge the j_{th} cluster to the i_{th}
6: end for
7: for iter in n
8: find the sequence sample in *iter* row of C by Equation (10) and mark it as $seq_{ct_{iter}}$
9: end for
Output: n sequences as centers $Cen\{seq_{ct_1}, \ldots, seq_{ct_n}\}$

3.3. Joint Modeling with Sequence Processing Networks

In the framework proposed herein, sequence data are initially processed using multivariate Dynamic Time Warping (mDTW), followed by clustering of the processed data to achieve a coarse categorization. Subsequently, the results of this coarse categorization are utilized as conditional inputs along with the sequence data (SEQ) for processing by the Conditional Gated Recurrent Unit (cGRU). The role of the cGRU is to enable the model to consider additional conditional or contextual information, therefore enhancing adaptability and performance for specific tasks. In this model, the additional condition provided is the coarse categorization result obtained from the initial clustering, while the sequence data itself offers contextual information. The processed data are then fed into a fully connected neural network, where fine categorization is performed using the SoftMax function, resulting in the final categorization outcomes after balancing the loss. As shown in Figure 4, the formulas for cGRU are as follows:

$$d_t = \tanh(W_d d_t + b_d) \tag{11}$$

$$s_t = \tanh(W_s s_t + b_s) \tag{12}$$

$$r_t = \sigma(W_r h_{t-1} + U_r d_t + V_r s_t + M_r fc + b_r) \tag{13}$$

$$z_t = \sigma(W_z h_{t-1} + U_z d_t + V_z s_t + M_z fc + b_z) \tag{14}$$

$$\tilde{h}_t = \tanh(W(r_t \odot h_{t-1}) + U d_t + V s_t + M fc + b) \tag{15}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t \tag{16}$$

$$o_t = \tanh(W_o h_t + U_o d_t + V_o s_t + M_o fc + b_o) \tag{17}$$

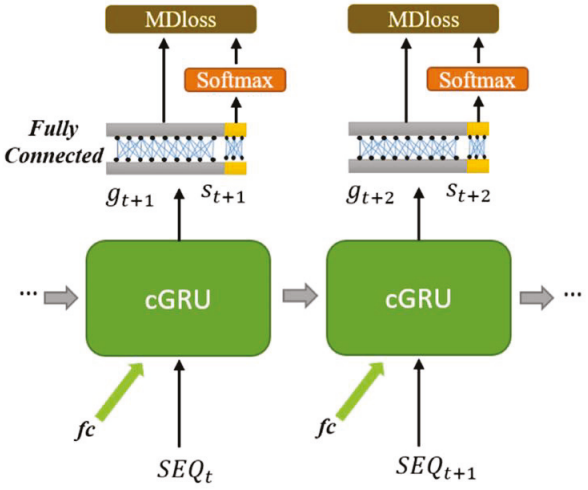


Figure 4. Detailed presentation of the core parts of the framework.

The variables utilized in Equations (11)–(17) are defined as follows:W, U, V, and M represent weight matrices, which are crucial parameters that govern the influence of various inputs on the neurons within a neural network layer. b denotes the biases, which are added to the weighted sum of inputs to adjust the output of the activation function. h_t signifies the hidden state at time t, encapsulating the internal representation of the network based on the inputs received up until that point. o_t represents the output vector at time t, which is generated by the network after processing the input and hidden state. The input vectord_t corresponds to the t-th state of the writing motion path signature, denoted as $\{d_{1,i}, d_{2,i}\}$. Here, i varies from 1 to n, indicating that the signature is comprised of a sequence of n points. s_t represents the pen's up–down state at time t, indicating whether the pen is touching the writing surface or not. The variable fc stands for the conditional input, which provides supplementary information to the network to influence its output.

The output of the conditional Gated Recurrent Unit (cGRU), o_t, is subsequently fed into two fully connected layers. These layers further process the output, ultimately yielding the system's output, which is a $M \times 5 + 3$ vector denoted as g_{t+1}, s_{t+1}. Here, g_{t+1} is a vector of size $M \times 5$, which serves as the input data for a Gaussian Mixture Model (GMM). The GMM is employed to predict pen movements, considering the probability distribution of potential movements. s_{t+1} is a 3×1 vector that is used for predicting the pen tip state, indicating its orientation or other pertinent properties. To enhance the performance of our networks, we utilize a Mixture Density Loss function (MDLoss). This loss function is specifically tailored to handle the output of mixture models, such as the GMM used in this context. By minimizing the MDLoss, we aim to refine the accuracy of the predicted pen movements and pen tip states, ultimately bolstering the overall performance of the system in generating realistic and precise handwriting simulations.

Unlike CGRU, other neural networks we utilize, such as LSTM or Transformer, do not possess a designated conditional input c. To address these discrepancies, we employ a method where we concatenate the output probability distribution of mDTW with the normalized original sequence information before inputting it into the LSTM or Transformer model, as listed in Equation (18)

$$i_t = concat(seq_t, c) \tag{18}$$

Among the many methods of time-series analysis and pattern recognition, dynamic time warping (DTW) and its multivariate version (mDTW) have proven to be effective tools, especially when dealing with time series of unequal lengths and varying speeds. However, traditional mDTW methods may be limited in handling large-scale, high-dimensional, and complex pattern data. In recent years, deep-learning models, such as Long Short-

Term Memory Networks (LSTM) and Transformer, have achieved remarkable success in processing sequence data. These models can automatically extract useful features from raw data, allowing them to handle complex patterns and long-term dependencies. To combine the flexibility of mDTW with the powerful representation capabilities of deep-learning models, we propose a joint modeling method. The core idea of this method is to embed mDTW as a differentiable module into the deep-learning framework so that the parameters of mDTW and the parameters of the deep-learning model can be optimized during the end-to-end training process. In the process of joint modeling, model selection is a crucial step because it will directly affect the model's ability to handle specific tasks. For the problem we face, namely the analysis of time-series data, we need to choose among multiple deep-learning models. Long short-term memory networks (LSTM) have proven to be very effective in processing time-series data and are able to capture long-term dependencies in the data, which is crucial for many time-series tasks. However, the Transformer model's advantages in handling parallelization and self-attention mechanisms make it more efficient when processing large-scale data sets and capable of capturing complex interactions between different parts of the input sequence. After selecting a suitable deep-learning model, the next step is to embed the mDTW algorithm into this model. The key to this step is to ensure that mDTW can be combined with the deep-learning model in a differentiable way so that we can use the backpropagation algorithm to train the entire model. To achieve this, we need to make appropriate modifications to the mDTW algorithm so that the gradients generated during the calculation process can be effectively passed to the deep-learning model. In this way, the deep-learning model can update its parameters based on the gradient information provided by mDTW, therefore achieving more efficient processing of time-series data.

In the model training phase, we need to design a joint loss function that will consider both the distance metric of mDTW and the classification or regression loss of the deep-learning model. In this way, we can ensure that the parameters of the mDTW algorithm and the deep-learning model are optimized simultaneously during the training process. This joint optimization method will enable the entire model to take into account the local shape characteristics of the data and capture the global structural information of the data when processing time-series data.

$$L = -\sum_{i=1}^{N} y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \tag{19}$$

In the equation, L is the value of the loss function, N is the number of samples, y_i is the actual label of the ith sample, p_i is the predicted probability that the ith sample belongs to the positive class (with label 1), and log denotes the natural logarithm.

Finally, in terms of model optimization, we will use appropriate optimization algorithms to update the model parameters. The gradient descent method or its variants are commonly used optimization algorithms that can adjust the parameters of the model based on the gradient information of the loss function, therefore gradually reducing the training error. In this process, we need to choose appropriate learning rates and other hyperparameters to ensure the stability and efficiency of the optimization process. Through continuous iteration and optimization, we will eventually obtain a joint model that cannot only utilize the flexibility of mDTW to process time-series data but also leverage the powerful representation capabilities of deep-learning models.

4. Experiments

Three experiments' results are shown in this paper. The first is a system of point matching for temporal sequences; the second is a system of online character recognition; The third one is human activity recognition. These experiments correspond to the three major components in our algorithm: mDTW, hierarchical cluster, and sequence processing performance.

4.1. Non-Rigid Sparse Point Matching

It is important to introduce the data we utilized. CASIA-HWDB, with its high-resolution color images and well-defined binary representation, stands as a valuable resource for offline handwritten character recognition research.

Point matching constitutes a classical and intricate challenge within the expansive domain of image registration research. When juxtaposed with the task of matching feature points in images, the endeavor of aligning points within handwritten characters introduces three distinctive characteristics that significantly shape the complexity of the problem. There is the imposition of a nonlinear spatial transformation when attempting to match points in handwriting characters. This nonlinear transformation induces a notable disturbance in the distribution of the two sets of matching points, introducing intricacies that are not present in linear transformations. This spatial nonlinearity poses a formidable obstacle, demanding sophisticated techniques to navigate and effectively align the points within the given framework. Especially as depicted in Figure 5, the significance of the point-matching process becomes strikingly evident when confronted with the notably distinct writing styles of various authors for the same character. This figure illustrates a scenario where two different individuals have written the same character, yet their unique writing styles result in significantly different representations. The red and blue line segments in the figure represent two online handwritten datasets, which are handwritten text samples that include time information. These datasets capture the intricate details of each author's writing style, encompassing the speed, pressure, and direction of the pen strokes. The green line segments indicate the point alignment relationships calculated using the modified Dynamic Time Warping (mDTW) algorithm. This algorithm allows us to match corresponding points between the two datasets, even when they exhibit different lengths and writing styles. Upon examining the point alignment process, it is evident that our results are highly accurate, as the green line segments closely adhere to the contours of the red and blue line segments, indicating a remarkable degree of similarity between the matched points. This underscores the effectiveness of our approach in addressing the challenges posed by diverse writing styles and emphasizes the crucial role of accurate point matching in online handwritten text recognition.

The nature of handwriting introduces an idiosyncratic challenge: individuals often connect different strokes for the sake of convenience during the writing process. This tendency results in a considerable number of errors in the sets of matching points. The interconnection of strokes complicates the matching task, requiring an intelligent and adaptable approach to correctly align points despite these inherent irregularities.

The feature points of Chinese characters exhibit a geometric dispersion that adds an additional layer of difficulty to the matching process. Unlike more regularly structured shapes, the geometric dispersion of feature points in Chinese characters makes it challenging to rely on local spatial distribution similarity for effective point matching. Consequently, conventional methods that hinge on local spatial characteristics may fall short when confronted with the intricacies of matching points within these dispersed geometric patterns. Point matching in handwriting characters is distinguished by the introduction of nonlinear spatial transformations, the prevalence of stroke connections leading to errors, and the geometric dispersion of feature points in Chinese characters. Addressing these challenges requires the development and application of advanced methodologies that can adeptly handle the intricacies inherent in the matching process within this unique context.

While point-matching is not the central focus of our investigation, its significance lies in its capacity to serve as a reflective indicator of the performance of mDTW. An extensive exploration of point-matching experiments has been conducted to comprehensively assess the prowess of mDTW. The findings reveal that mDTW takes into account various aspects such as handwriting trajectory, direction distribution, moving displacement, and stroke connection in tandem to determine the outcomes of the matching process. A detailed analysis of the experimental results, as depicted in the figures, allows for an intuitive description of the prevailing scenario wherein correctly matched points dominate the

major portions of the matching results. The proposed mDTW algorithm has effectively met our stipulated requirements, demonstrating a reliable point-matching performance essential for subsequent clustering endeavors. The amalgamation of trajectory, distribution, displacement, and stroke considerations in the mDTW framework contributes to its efficacy in delivering accurate and meaningful matching outcomes, therefore substantiating its suitability for the specified clustering objectives. To give a further analysis and evaluation on mDTW, we compare our method with two traditional Chinese handwriting points matching methods [22,23]. These two algorithms are image-based. In other words, they do not consider temporal information while matching. Similar to some other Chinese handwriting research, in addition, we also compared two of the latest point-matching methods, LVMs [24] and PMT [25]. LVMs, or Large Vision Models, leverage semantic features derived from large-scale vision models to enhance geometry-based shape feature learning, making them highly effective for non-rigid point cloud matching. PMT, or Proxy Match Transform, introduces a high-order feature transform layer to efficiently match geometric shapes in 3D point clouds, particularly excelling in assembling geometric shapes by establishing reliable local correspondences between part surfaces. We divide the sequence samples into three groups by their complexities, which are simple (0–10 points), normal (20–50 points) and complex (more than 50) points. The summary statistics for comparison are illustrated in Table 1. APM refers to Asymmetric Point Matching, while CGE stands for Constrained Global Energy function.

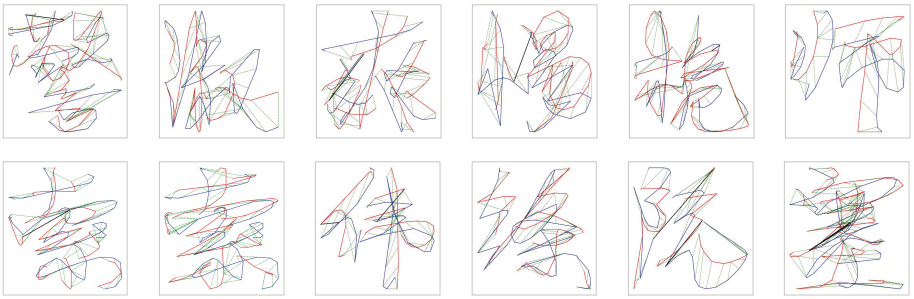


Figure 5. Point-matching results, black line indicates wrong, green indicates right. Red and blue lines are two handwriting characters.

We randomly selected 200 pairs of Chinese handwriting samples from the OLHWDB1.1 dataset. In this dataset, the shortest sample contains 3 points, while the longest one has 216 points. OLHWDB (Online Handwritten Chinese Character Database) is the online handwritten version of HWDB (Handwritten Chinese Character Database), capturing handwritten point sequences with both spatial and temporal information. In contrast, HWDB represents characters in image format and lacks sequential data. The OLHWDB1.1 dataset comprises 3755 distinct Chinese characters, each represented by numerous handwriting samples. For our point registration experiment, we randomly selected 200 of these characters to ensure a diverse and representative subset for analysis. Since the purpose of this experiment is to demonstrate the accuracy of point registration, it is not necessary to select all Chinese characters. The experiment data are labeled and checked manually. Results reported in Table 2 indicate that the proposed mDTW has an advantage against state-of-the-art methods in all complexity conditions. A possible explanation for this is that mDTW takes temporal information as a significant factor. Since online handwriting trajectories may have severe spatial noises in different samples, the writing order for every stroke should be the same. Thus, sequence-based mDTW performs much better than image-based methods in such kinds of tasks.

Table 1. Comparison of point matching results.

Method	Simple	Normal	Complex
APM [22]	34.2%	73.4%	82.7%
CGE [23]	87.1%	91.2%	85.3%
LVMs [24]	90.5%	92.0%	88.7%
PMT [25]	89.8%	93.5%	87.4%
mDTW	93.1%	94.2%	95.7%

4.2. Handwriting Character Recognition Systems

We present the settings of the model hyperparameters to facilitate reproducibility. For the mDTW+Transformer model, we employ a 6-layer Transformer architecture to efficiently process temporal sequence data, leveraging the modified Dynamic Time Warping (mDTW) algorithm to enhance its capability to handle sequences with varying lengths and temporal distortions. Each layer of the Transformer is designed with a hidden state space dimension of 512. This configuration is crucial for balancing computational efficiency with the model's ability to capture complex temporal relationships. The choice of six layers is informed by preliminary experiments that indicate this depth provides a suitable trade-off between model complexity and performance on our targeted tasks without leading to significant overfitting or computational inefficiency. The hidden state dimension of 512 for each layer was selected based on its performance in capturing the intricacies of the data while maintaining a manageable model size and computational demand. These hyperparameters—specifically, the depth of the Transformer and the dimensionality of the hidden states—are essential components of our model's architecture that significantly influence its performance and efficiency in processing time-series data.

The ultimate proposal of this paper is the HCCR experiment. We compare our method with state-of-the-art methods, including Benchmark, ICDAR-2011 Winner: VO-3, UWarwick, DropSample and DropSample-Ensemble-9 [26], NET4-subseq30 and NET123456 [27] and RNN-attention [19]. The illustrated results of those methods are recorded from the corresponding reference. In Table 2, "top-1", "top-3", and "top-10" refer to metrics for model prediction accuracy. "Top-1" accuracy means the proportion of times the model's first choice is the correct answer; "top-3" accuracy indicates the chance the correct answer is within the model's top three predictions; similarly, "top-10" accuracy refers to the correct answer being among the model's top ten predictions.

As can be seen from Table 2, the result of mDTW+Transformer architecture is the current best performance among state-of-the-art methods. It should be mentioned that the results we utilize to compare are from all deep-learning-based end-to-end methods. Our method fails in the top-1 test against other state-of-the-art since we used the mDTW-cluster model to do the class task, which means that a test sample only refers to the nearest centers. Nevertheless, the top-3 result shows the achievements our method has made, indicating that the test samples find the correct centers as top-3 in 3755 classes. All these experiment outcomes indicate that our method has a huge potential in sequence processing tasks. We then combined it with the optimized Transformer to form the mDTW+Transformer architecture. As can be seen, this new architecture integrates the strengths of both components, surpassing the accuracy of all other compared SOTA methods.

In recent times, there have been notable strides in the realm of sequence modeling, particularly with the advancements achieved by RNN-attention-based algorithms. These methodologies, leveraging attention mechanisms, have demonstrated significant progress across various domains. A noteworthy example is the RNN:Ensemble [19], which effectively reduced model scale while maintaining competitive performance. However, upon scrutinizing the data presented in Table 2, it becomes apparent that deep-learning-based models appear to have reached a performance plateau. The observed trend indicates that new methodologies have only marginally improved accuracy, with gains typically falling

below the threshold of 0.3%. Despite the collective efforts to enhance these models, the incremental improvements have been relatively modest.

Table 2. Comparison of HCCR methods.

Methods	Top-1	Top-3	Top-10
Benchmark 2013 [18]	95.3%	97.3%	98.8%
VO-3 2013 [28]	95.8%	97.8%	99.0%
UWarwick 2013 [28]	97.4%	98.4%	99.3%
Runner-up 2013 [28]	96.9%	98.2%	99.2%
DropSample 2016 [26]	97.2%	98.4%	99.4%
Ensemble-9 2016 [26]	97.5%	98.5%	99.5%
NET4-subseq30 2017 [27]	97.9%	98.9%	99.5%
NET123456 2017 [27]	98.1%	99.0%	99.5%
RNN:Single 2019 [19]	97.2%	98.4%	99.4%
RNN:Ensemble 2019 [19]	97.6%	98.6%	99.5%
MSCS+ASA+SCL 2022 [29]	96.7%	97.6%	98.9%
SqueezeNext+CCBAM10 2023 [29]	97.4%	98.6%	99.2%
DLHCR 2024 [29]	97.8%	98.7%	99.4%
mDTW-Cluster	68.7%	88.8%	94.6%
mDTW+cGRU	97.4%	99.1%	99.6%
mDTW+Transformer	98.5%	99.3%	99.8%

Breaking away from this performance bottleneck, our proposed method has significantly altered the landscape. Our approach introduces a paradigm shift, outperforming existing models and achieving a notable breakthrough in accuracy. This transformative outcome underscores the potential for further advancements in sequence modeling, challenging the limitations imposed by the perceived performance ceiling in deep-learning-based methodologies. Except for higher HCCR accuracy, the proposed one has an advantage in the real meanings of model parameters when compared to these state-of-the-art methods. For instance, in this paper, the proposed model is a sequence group, where each sequence stands at the center of the character. On the contrary, parameters in deep-learning-based models are weights and biases. It is very hard to figure out the relationships between these parameters and real samples in a black box model.

4.3. Human Activity Recognition Experiments

We present the experimental evaluation of our proposed method, mDTW+Transformer, on three human activity datasets: UCI-HAR, PAMAP2, and OPPORTUNITY. The datasets were chosen due to their popularity and representative nature in the field of human activity recognition. The experiments aim to demonstrate the effectiveness of our approach in comparison to the state-of-the-art (SOTA) methods. For the experimental evaluation, we followed a standard protocol for data preprocessing, which is borrowed from [30]. We compared our proposed method, mDTW+Transformer, with several SOTA methods on each dataset. The results of our experiments are summarized in Tables 1–3 for point matching and HCCR, respectively. In Table 3, the hyphens indicate that the given algorithm was not tested on the specific dataset.

From the tables, it is evident that our proposed method, mDTW+Transformer, outperforms all the compared SOTA methods on all datasets in terms of accuracy. The results demonstrate the effectiveness of our approach in recognizing human activities from various datasets. The superior performance of our method can be attributed to several factors: (i) the combination of mDTW and Transformer allows for a robust alignment between different activity instances, (ii) the Transformer architecture captures long-range dependencies effectively, and (iii) the use of appropriate evaluation metrics ensures a fair comparison with SOTA methods. Furthermore, our method is generic and can be applied to various

human activity recognition tasks without any modifications, making it a versatile approach for practical applications.

Table 3. Comparison of HAR methods.

Methods	UCI-HAR	WISDM	PAMAP2	UniMib	OPPOR
DCNN 2015 [31]	95.1%	-	-	-	-
tFFT+Convnet 2016 [32]	95.7%	-	-	-	-
b-LSTM-S 2016 [33]	-	-	-	-	92.7%
DeepConvLSTM 2016 [34]	-	-	-	-	93.0%
CondConv 2018 [35]	-	98.7%	94.0%	77.31%	81.1%
CIRF 2018 [36]	-	-	-	-	89.1%
LSTM+CT+CSA 2018 [37]	-	-	89.96%	-	-
AttnSense 2019 [38]	-	-	89.3%	-	-
The Layer-wise CNN 2020 [39]	96.9%	98.8%	92.9%	78.0%	81.0%
Lego CNN 2021 [40]	91.4%	97.5%	93.5%	74.4%	88.1%
Real+Virtual 2023 [41]	-	-	69.9%	-	-
TinyTNAS 2024 [42]	96.7%	96.5%	93.7%	77.9%	93.2%
mDTW+Transformer	88.7%	98.5%	94.2%	78.5%	93.5%

Our experimental results with the mDTW+Transformer method outperform the state-of-the-art (SOTA) due to a key characteristic of our proposed approach: the coarse-to-fine classification method applied to sequences. The results are listed in Table 3. This method involves a two-step classification process that enhances both the accuracy and efficiency of classification.

In the first step, coarse-grained classification, sequences are initially grouped into broad categories based on their overall features and patterns. This initial stage focuses on capturing the overall characteristics of the sequences, reducing the complexity of subsequent processing. In the second step, fine-grained classification, sequences within each broad category are further classified with greater precision. This stage emphasizes the subtle features and patterns within sequences, aiming to improve the accuracy of classification. By employing this coarse-to-fine classification strategy, our mDTW+CGRU method comprehensively considers the features of sequences and effectively captures complex patterns. Additionally, the CGRU architecture, with its powerful expression capabilities, effectively learns long-term dependencies within sequences, further enhancing the performance of classification. Therefore, we attribute the superior experimental results of our mDTW+CGRU method compared to SOTA to the coarse-to-fine classification approach. This strategy not only improves classification accuracy but also better adapts to the complexity of various sequence data.

4.4. Ablation Study

The target of Experiment 2 was to investigate the effect of model hyperparameters on different complexity of characters. We manually selected 100 handwriting sample couples, including 33 simple, 34 normal, and 33 complex. We investigated 7 groups of hyperparameters, where we only change α, β is configured as a constant vector $\{1, 10, 100\}$. Table 4 compares the data we obtained from experiments. It could be concluded that the value of α_2 plays a significant role in our algorithm. On the contrary, the other two α variables are not key factors. Meanwhile, appropriate gaining of α_1 and α_3 could increase matching accuracy. Considering that the normal complexity is a major part of all characters, we set α as $\{0.1, 0.8, 0.1\}$ for clustering. The challenge of data significance arises from the inherent properties of diverse variables. One particularly well-established feature in the realm of handwriting analysis research is the pen's moving displacement, denoted as Δx. This variable has gained widespread recognition for its efficacy in capturing important variations in writing directions. Researchers have extensively employed it to unveil nuanced aspects of handwriting intricacies.

Table 4. Results of the effect of different model hyperparameters α on different complexity of characters.

α	Simple	Normal	Complex
$\{0, 1, 0\}$	89.5%	92.2%	89.3%
$\{0.1, 0.8, 0.1\}$	93.1%	94.2%	95.7%
$\{0.45, 0.45, 0.1\}$	66.3%	42.2%	32.7%
$\{0.8, 0.1, 0.1\}$	50.3%	62.2%	48.6%
$\{0.1, 0.45, 0.45\}$	43.6%	38.1%	30.2%
$\{0.45, 0.1, 0.45\}$	56.7%	18.8%	21.9%
$\{0.1, 0.1, 0.8\}$	68.0%	18.2%	15.3%

However, it is essential to delve into the relationship between pen-moving displacement and pen location. The latter can be conceptualized as the integral of displacement, offering a holistic perspective on the trajectory of the writing instrument. Yet, a notable drawback surfaces: the directional variation information tends to be subdued within the integral representation. Consequently, while pen location provides an overall picture of the writing path, it may lack the granularity required to discern subtle changes in direction. In stark contrast, the utilization of pen-moving displacement emerges as a superior choice in many instances. By focusing on the instantaneous changes in position, this approach excels at capturing the finer nuances of writing directions. The importance of these variations becomes particularly evident when seeking a comprehensive understanding of the intricate dynamics involved in handwriting.

We conducted ablation experiments to validate the effectiveness of our method, experimenting on both the HCCR dataset and the Human Action Recognition dataset. In these experiments, we selected three sequence modeling models: cGRU, LSTM, and Transformer. We first conducted experiments using these three models individually, and then we integrated them into the mDTW framework. By comparing and observing the impact of the mDTW framework on the accuracy of the experimental results, it can be seen from Table 5 that the experimental results with mDTW achieved an accuracy improvement of about 1% on most datasets. This demonstrates the effectiveness of our proposed method.

Table 5. Ablation of mDTW Joint modeling methods.

Methods	HCCR	UCI-HAR	WISDM	PAMAP2	UniMib	OPPOR
cGRU	95.6%	84.1%	94.8%	92.9%	75.3%	88.1%
LSTM	95.1%	86.3%	94.5%	93.7%	75.8%	89.6%
Transformer	96.6%	86.2%	95.5%	93.2%	76.4%	90.4%
mDTW+cGRU	97.4%	87.6%	97.1%	94.1%	77.1%	92.1%
mDTW+LSTM	97.2%	88.1%	97.9%	94.1%	77.7%	92.7%
mDTW+Transformer	98.5%	88.7%	98.5%	94.2%	78.5%	93.5%

In summary, the preference for pen-moving displacement over pen location stems from its ability to accentuate crucial directional changes, therefore outperforming the integral representation. This nuanced perspective enables a more detailed and insightful analysis of the complexities inherent in the act of writing.

5. Conclusions

The purpose of the current study was to develop a sequence algorithm for temporal data modeling applications. To achieve this, we proposed a simplified hierarchical cluster model and designed a mDTW method to value the similarities between sequences. By taking HCCR as a specific research area, the proposed method has achieved a big improvement in recognizing accuracy (Top-3 and Top-10). This work contributes great confidence that reforming, improving, and appropriately using some simple methods could defeat deep learning in large-scale data-sequence processing tasks. We can outperform deep-learning approaches in the realm of intelligent computing for large-scale data-sequence processing

tasks. Moreover, we envision significant potential for the expansion of our method into other research domains focused on sequence analysis within the purview of intelligent computation. Additionally, the methodology presented in this article seamlessly lends itself to joint modeling with various deep neural network models, further enhancing its adaptability and effectiveness in intelligent computing applications. Deep models include main sequence-oriented methods, including RNN, Tf, and other models. Experimental results show that our method can improve the modeling performance of the model to a certain extent when combined with a deep model. It has achieved good performance in human action recognition and handwriting recognition tasks, and some results exceed SOTA.

Through joint modeling, we can effectively combine the flexibility of mDTW with the powerful representation capabilities of deep-learning models. This method has significant advantages when dealing with complex time-series analysis tasks, especially when the shape and global structure of the time series need to be considered simultaneously. Future work will include validating the effectiveness of this joint modeling approach in a wider range of application scenarios and exploring further optimization strategies to improve model performance and efficiency.

Author Contributions: Conceptualization, B.S. and C.C.; methodology, B.S.; software, B.S.; validation, C.C.; formal analysis, B.S.; investigation, B.S.; resources, C.C.; data curation, B.S.; writing—original draft preparation, B.S.; writing—review and editing, C.C.; visualization, B.S.; supervision, C.C.; project administration, C.C.; funding acquisition, C.C. All authors have read and agreed to the published version of the manuscript.

Funding: This paper is sponsored by the General Program of the National Natural Science Foundation of China (No. 62376279).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: https://nlpr.ia.ac.cn/databases/handwriting/Online_database.html.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cai, X.; Xu, T.; Yi, J.; Huang, J.; Rajasekaran, S. DTWNet: A Dynamic Time Warping Network. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 11640–11650.
2. Ismail Fawaz, H.; Forestier, G.; Weber, J.; Idoumghar, L.; Muller, P.A. Deep learning for time series classification: A review. *Data Min. Knowl. Discov.* **2019**, *33*, 917–963. [CrossRef]
3. Qu, Y.; Yang, M.; Zhang, J.; Xie, W.; Qiang, B.; Chen, J. An outline of multi-sensor fusion methods for mobile agents indoor navigation. *Sensors* **2021**, *21*, 1605. [CrossRef] [PubMed]
4. Luo, Z.; Qi, R.; Li, Q.; Zheng, J.; Shao, S. ABODE-Net: An Attention-based Deep Learning Model for Non-intrusive Building Occupancy Detection Using Smart Meter Data. In Proceedings of the International Conference on Smart Computing and Communication, New York, NY, USA, 18–20 November 2022; pp. 152–164.
5. Song, C.; Lu, M.; Wang, Y.; Lu, W. A dynamic time warping loss-based closed-loop CNN for seismic impedance inversion. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 5925313. [CrossRef]
6. Middlehurst, M.; Schäfer, P.; Bagnall, A. Bake off redux: A review and experimental evaluation of recent time series classification algorithms. *Data Min. Knowl. Discov.* **2024**, *38*, 1958–2031. [CrossRef]
7. Shen, J.; Bao, S.D.; Yang, L.C.; Li, Y. The PLR-DTW method for ECG based biometric identification. In Proceedings of the 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Boston, MA, USA, 30 August–3 September 2011; pp. 5248–5251.
8. Schirrmeister, R.T.; Springenberg, J.T.; Fiederer, L.D.J.; Glasstetter, M.; Eggensperger, K.; Tangermann, M.; Hutter, F.; Burgard, W.; Ball, T. Deep learning with convolutional neural networks for EEG decoding and visualization. *Hum. Brain Mapp.* **2017**, *38*, 5391–5420. [CrossRef]
9. Lerogeron, H.; Picot-Clémente, R.; Rakotomamonjy, A.; Heutte, L. Approximating dynamic time warping with a convolutional neural network on EEG data. *Pattern Recognit. Lett.* **2023**, *171*, 162–169. [CrossRef]
10. Kate, R.J. Using dynamic time warping distances as features for improved time series classification. *Data Min. Knowl. Discov.* **2016**, *30*, 283–312. [CrossRef]

11. Zhang, H.; Dong, Y.; Li, J.; Xu, D. An efficient method for time series similarity search using binary code representation and hamming distance. *Intell. Data Anal.* **2021**, *25*, 439–461. [CrossRef]
12. Gold, O.; Sharir, M. Dynamic Time Warping and Geometric Edit Distance: Breaking the Quadratic Barrier. *ACM Trans. Algorithms* **2016**, *14*, 50. [CrossRef]
13. Ibrahim, M.Z.; Mulvaney, D. Geometry based lip reading system using multi dimension dynamic time warping. In Proceedings of the 2012 Visual Communications and Image Processing, San Diego, CA, USA, 27–30 November 2012; pp. 1–6.
14. Gupta, N.; Gupta, S.K.; Pathak, R.K.; Jain, V.; Rashidi, P.; Suri, J.S. Human activity recognition in artificial intelligence framework: A narrative review. *Artif. Intell. Rev.* **2022**, *55*, 4755–4808. [CrossRef] [PubMed]
15. Ramanujam, E.; Perumal, T.; Padmavathi, S. Human activity recognition with smartphone and wearable sensors using deep learning techniques: A review. *IEEE Sensors J.* **2021**, *21*, 13029–13040. [CrossRef]
16. Lockhart, J.W.; Pulickal, T.; Weiss, G.M. Applications of mobile activity recognition. In Proceedings of the 2012 ACM Conference on Ubiquitous Computing, Pittsburgh, PA, USA, 5–8 September 2012; pp. 1054–1058.
17. Vaizman, Y.; Ellis, K.; Lanckriet, G. Recognizing detailed human context in the wild from smartphones and smartwatches. *IEEE Pervasive Comput.* **2017**, *16*, 62–74. [CrossRef]
18. Liu, C.L.; Yin, F.; Wang, D.H.; Wang, Q.F. Online and offline handwritten Chinese character recognition: Benchmarking on new databases. *Pattern Recognit.* **2013**, *46*, 155–162. [CrossRef]
19. Ren, H.; Wang, W.; Liu, C. Recognizing online handwritten Chinese characters using RNNs with new computing architectures. *Pattern Recognit.* **2019**, *93*, 179–192. [CrossRef]
20. Zhang, X.Y.; Bengio, Y.; Liu, C.L. Online and offline handwritten Chinese character recognition: A comprehensive study and new benchmark. *Pattern Recognit.* **2017**, *61*, 348–360. [CrossRef]
21. Li, Z.; Xiao, Y.; Wu, Q.; Jin, M.; Lu, H. Deep template matching for offline handwritten Chinese character recognition. *J. Eng.* **2020**, *2020*, 120–124. [CrossRef]
22. Lian, W.; Zhang, L.; Yang, M. An Efficient Globally Optimal Algorithm for Asymmetric Point Matching. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1281–1293. [CrossRef]
23. Zhao, B.; Yang, M.; Pan, H.; Zhu, Q.; Tao, J. Nonrigid point matching of Chinese characters for robot writing. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, China, 5–8 December 2017; pp. 762–767.
24. Chen, Z.; Jiang, P.; Huang, R. Unsupervised Non-Rigid Point Cloud Matching through Large Vision Models. *arXiv* **2024**, arXiv:2408.08568.
25. Lee, N.; Min, J.; Lee, J.; Kim, S.; Lee, K.; Park, J.; Cho, M. 3D Geometric Shape Assembly via Efficient Point Cloud Matching. *arXiv* **2024**, arXiv:2407.10542.
26. Yang, W.; Jin, L.; Tao, D.; Xie, Z.; Feng, Z. DropSample: A new training method to enhance deep convolutional neural networks for large-scale unconstrained handwritten Chinese character recognition. *Pattern Recognit.* **2016**, *58*, 190–203. [CrossRef]
27. Zhang, X.Y.; Yin, F.; Zhang, Y.M.; Liu, C.L.; Bengio, Y. Drawing and recognizing chinese characters with recurrent neural network. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 849–862. [CrossRef] [PubMed]
28. Yin, F.; Wang, Q.F.; Zhang, X.Y.; Liu, C.L. ICDAR 2013 Chinese handwriting recognition competition. In Proceedings of the 2013 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013; pp. 1464–1470.
29. Kriuk, B.; Kriuk, F. Deep Learning-Driven Approach for Handwritten Chinese Character Classification. *arXiv* **2024**, arXiv:2401.17098.
30. Wang, X.; Zhang, L.; Huang, W.; Wang, S.; Wu, H.; He, J.; Song, A. Deep convolutional networks with tunable speed–accuracy tradeoff for human activity recognition using wearables. *IEEE Trans. Instrum. Meas.* **2021**, *71*, 2503912. [CrossRef]
31. Jiang, W.; Yin, Z. Human activity recognition using wearable sensors by deep convolutional neural networks. In Proceedings of the 23rd ACM International Conference on Multimedia, Brisbane, Australia, 26–30 October 2015; pp. 1307–1310.
32. Ronao, C.A.; Cho, S.B. Human activity recognition with smartphone sensors using deep learning neural networks. *Expert Syst. Appl.* **2016**, *59*, 235–244. [CrossRef]
33. Hammerla, N.Y.; Halloran, S.; Plötz, T. Deep, convolutional, and recurrent models for human activity recognition using wearables. *arXiv* **2016**, arXiv:1604.08880.
34. Ordóñez, F.J.; Roggen, D. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors* **2016**, *16*, 115. [CrossRef]
35. Ignatov, A. Real-time human activity recognition from accelerometer data using Convolutional Neural Networks. *Appl. Soft Comput.* **2018**, *62*, 915–922. [CrossRef]
36. Hu, C.; Chen, Y.; Hu, L.; Peng, X. A novel random forests based class incremental learning method for activity recognition. *Pattern Recognit.* **2018**, *78*, 277–290. [CrossRef]
37. Zeng, M.; Gao, H.; Yu, T.; Mengshoel, O.J.; Langseth, H.; Lane, I.; Liu, X. Understanding and improving recurrent networks for human activity recognition by continuous attention. In Proceedings of the 2018 ACM International Symposium on Wearable Computers, Singapore, 8–12 October 2018; pp. 56–63.
38. Ma, H.; Li, W.; Zhang, X.; Gao, S.; Lu, S. AttnSense: Multi-level Attention Mechanism For Multimodal Human Activity Recognition. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, Macao, China, 10–16 August 2019; pp. 3109–3115. [CrossRef]

39. Teng, Q.; Wang, K.; Zhang, L.; He, J. The layer-wise training convolutional neural networks using local loss for sensor-based human activity recognition. *IEEE Sensors J.* **2020**, *20*, 7265–7274. [CrossRef]
40. Tang, Y.; Teng, Q.; Zhang, L.; Min, F.; He, J. Layer-wise training convolutional neural networks with smaller filters for human activity recognition using wearable sensors. *IEEE Sensors J.* **2020**, *21*, 581–592. [CrossRef]
41. Leng, Z.; Kwon, H.; Plötz, T. Generating virtual on-body accelerometer data from virtual textual descriptions for human activity recognition. In Proceedings of the 2023 ACM International Symposium on Wearable Computers, Cancun, Mexico, 8–12 October 2023; pp. 39–43.
42. Saha, B.; Samanta, R.; Ghosh, S.K.; Roy, R.B. TinyTNAS: GPU-Free, Time-Bound, Hardware-Aware Neural Architecture Search for TinyML Time Series Classification. *arXiv* **2024**, arXiv:2408.16535.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

Model for Determining the Psycho-Emotional State of a Person Based on Multimodal Data Analysis

Nataliya Shakhovska [1,2,*], Oleh Zherebetskyi [1] and Serhii Lupenko [3,4]

[1] Department of Artificial Intelligence, Lviv Polytechnic National University, 79013 Lviv, Ukraine
[2] College of Engineering, Design and Physical Sciences, Brunel University, London UB8 3PH, UK
[3] Faculty of Electrical Engineering, Automatic Control and Informatics, Opole University of Technology, 45758 Opole, Poland
[4] Institute of Telecommunications and Global Information Space, National Academy of Sciences of Ukraine, 02000 Kyiv, Ukraine
* Correspondence: nataliya.b.shakhovska@lpnu.ua

Abstract: The paper aims to develop an information system for human emotion recognition in streaming data obtained from a PC or smartphone camera, using different methods of modality merging (image, sound and text). The objects of research are the facial expressions, the emotional color of the tone of a conversation and the text transmitted by a person. The paper proposes different neural network structures for emotion recognition based on unimodal flows and models for the margin of the multimodal data. The analysis determined that the best classification accuracy is obtained for systems with data fusion after processing each channel separately and obtaining individual characteristics. The final analysis of the model based on data from a camera and microphone or recording or broadcast of the screen, which were received in the "live" mode, gave a clear understanding that the quality of the obtained results is highly dependent on the quality of the data preparation and labeling. This is directly related to the fact that the data on which the neural network is trained is highly qualified. The neural network with combined data on the penultimate layer allows a psycho-emotional state recognition accuracy of 0.90 to be obtained. The spatial distribution of emotion analysis was also analyzed for each data modality. The model with late fusion of multimodal data demonstrated the best recognition accuracy.

Keywords: multimodal data; late fusion; convolution neural network; emotional state; multi-modal emotion recognition

1. Introduction

More than a decade ago, the realm of human–machine interaction began gaining traction, progressing toward increasingly user-friendly and adaptable machine interfaces. With the emergence of machine learning, the capability to discern human emotions became feasible [1].

Examining interpersonal interaction underscores the importance of comprehending the psycho-emotional state of conversational partners. Consequently, the integration of emotion recognition into machine–human interaction processes has surged in popularity [2].

Traditionally, humans perceived machines as emotionless entities primarily executing intricate computations. However, equipping machines with the ability to decipher facial expressions, intonation, gestures, and other non-verbal cues promises enhanced rapport between machines and individuals [3].

Many global car brands often introduce systems that warn the driver about his drowsiness [4] or feeling unwell. Popular brands are also willing to pay money to evaluate human emotions while viewing advertisements [5]. A human behavior recognition system is often helpful in predicting crimes [6]. And, there are many other areas in which understanding the emotions of consumers would play the role of an essential push in development [7].

Some reports suggest that the emotion detection and recognition market was worth about $12 billion in 2018 and could grow to more than $90 billion by 2024 [8]. The Israeli startup Beyond Verbal (Tel Aviv-Yafo, Israel) [9] is developing intelligent emotion analytics that look for signs of anger, anxiety and excitement by analyzing a person's voice—not the specific words, but intonations. The proposed approach may be helpful for mental health assessments, market research or even to help call centers improve phone customer relations [10].

Amazon (Bellevue, WA, USA) is also working to improve the ability of its own Alexa AI assistant [11] to interact with users by detecting emotions in their voices. Reference [12] proposed the dataset MOISEI to examine a data fusion model. However, the first research on the multimodal analysis of human emotions was conducted based on two modal parts of information. For example, the usage of a combination of sound and speech modalities is analyzed in [13]. Also, an implementation with a combination of sound and speech in systems with a deep neural network is given in the article in [14]. Based on these works, it was concluded that the fusion of two modal signals provides better accuracy than any unimodal system.

There are also many algorithms combining different types of information signals into multimodal data at various stages (early, late) and using other methods of intermodal information extraction. All these implementations are not easy to understand, but they will eliminate the possible loss of information, which is the basis of multimodal emotion data, and give perfect accuracy.

At this stage of development, models are studied and improved based on intermodal information in multimodal recognition of the psycho-emotional state.

The research aims to develop a model and information system for recognizing human emotions in a stream obtained from a PC or smartphone camera, using different methods of merging channels of image, sound and text, and analysis of the results.

To achieve this goal, it is necessary to solve the following tasks:

- Live generation of data received from a camera and microphone;
- Live generation of data obtained from a recording in the device's memory;
- Live generation of data obtained from the broadcast screen;
- Processing live data to a form that neural networks will accept;
- A mechanism for generating new data to supplement ready-made data sets and create new ones;
- Three neural networks for separate processing of image, sound and text streams;
- Two neural networks for multimodal recognition of emotions;
- Three information systems for recognizing emotions with different ways of merging data streams;
- Interface for ease of use of information systems.

Having solved all these problems, we will be able to conduct a qualitative analysis of the three selected ways to merge data streams and draw appropriate conclusions.

The objectives of this research are:

- The formalization of the process of the multimodal data merging and analysis;
- Each data modality analysis and neural network architecture development;
- The development of the methods of late fusion of multimodal data to achieve the best recognition accuracy;
- Proving the adequacy and demonstration of the functionality of the created model.

The main contribution of the research is given below:

1. The deep learning-based late fusion of multimodal information for human emotion classification of sequential images, audio and text is developed. To achieve this, five neural network structures based on CNN and LSTM for single-mode and multimode emotion recognition are proposed. The number of emotions is one of the model's parameters and can be easily changed. Bidirectional LSTM with CNN where the first layer is wrapped with a TimeDistributed function is used for face emotion recognition.

2. The comparative simulation experiment of single-mode and multimode emotion recognition of emotional features in artificial intelligence information systems is conducted. The best accuracy in the multimodal emotion recognition (MMER) network is 0.9245 with merging in the end (late fusion). However, due to the low accuracy of speech emotion recognition (SER) and text emotion recognition (TER)—0.5611 and 0.64, respectively, this system shows high quality in recognition based on multimodal data obtained in the "live" mode.

The rest of the paper is organized in the following way: Section 2 outlines some recent studies and their analyses, including all the papers that use the dataset examined in this study; Section 3 starts with a detailed description of the data preparation and ways of data merging and follows with the developed model; Section 4 presents the obtained results; Section 5 is the related discussion; and Section 6 is the concluding section.

2. Related Works

Over time, systems began to use fusion at different stages, such as early (at the function level) and late (weighted average of unimodal network results). In [12], a method for processing three types of multimodal data is proposed, which considers the delay and time shift of multimodal data in the time dimension and the relationship between embedding in multimodal data. The evaluation is based on the MOSEI dataset. Experimental results show that the method effectively improves the correlation of relevant information between multimodal signals.

Combining features at the input level and combining deep neural networks for emotion analysis based on multimodal data is described in [15]. In [16,17], the author first encodes each modality separately and then uses feature-level fusion in the middle layer to obtain a multimodal embedding, which is a static feature-level fusion. In contrast to the method described in previous articles, the author in [18,19] uses dynamic fusion at the feature level, the essence of which is the simultaneous encoding of unimodal data and fusion at the feature level. Each information modality is given the same weight in the methods described above. However, this approach is only partially correct because during further research in articles [20–23], researchers found that different modal information has other importance for the final result, and the degree of importance differs. For example, in [23], the author noted that verbal data would be more influential than non-verbal data, leading to a change in the word's meaning. The author takes the verbal modality as dominant and the non-verbal modality as an auxiliary for multimodal fusion.

The papers [24,25] analyze separated modalities for mood detection and modality and fusion. The multimodal-fusion approach in [25] uses a strategy referred to as "late-fusion" that involves the combination of unimodal model outputs as inputs of the decision engine.

During the inception of this technology, there needed to be better quality of emotion determination. This problem has an explanation. Emotions reflect a person's subjective attitude to various phenomena. Emotion is a person's specific response to situations that affect his/her interests. There are emotional manifestations that cannot be controlled and that can be recognized. Under the influence of the same external factors, different people may have different emotions [26]. Emotions are manifested in different ways and in combination with each other. Recognizing emotions from facial expressions is a complex mental process. A person can recognize another person's emotions by their face, gait and gestures. Ekman, a psychologist in the 1960s, believed that there were only six basic emotions expressed by facial emotions [27,28]. But further research has shown much more variability regarding the number of emotional states and how they are expressed. This varies in different cultures [29], in different situations, and even within the same day.

One of the companies offered police software to detect emotions [30]. The ability to detect emotions such as anger, stress or anxiety provides law enforcement with additional information when conducting a large-scale investigation. Ultimately, it is believed that the responsible use of this technology will be a factor that will make the world safer.

Many job candidates have benefited from HireVue's technology, which aims to help eliminate the very significant human bias in the existing hiring process [31]. Cogito has developed voice analysis algorithms for call center personnel to help them detect when customers are experiencing problems. However, people believe that before emotion detection can make automated decisions, the industry needs more evidence that machines can detect human emotions effectively and consistently [32].

The field of recognition of a person's psycho-emotional state by a machine gained popularity with the advent of machine learning and neural networks. This is directly related to the fact that to perform this task, it is necessary to analyze the image by extracting specific features from it; the same should be performed with the data of other modalities, namely sound and text. Considering unimodal data, the first development stage was using simple fully connected neural networks. However, such data as image, sound, and text carry much more information if we consider not one pixel (for an image) but also its surroundings. This is characteristic of the next stage, namely convolution neural network (CNN) [33]. This neural network performed exceptionally well in emotion recognition based on all three data types, but separately.

However, as mentioned above, data for recognizing a person's psycho-emotional state should be taken with a specific perspective concerning time. For this, at the stage of development of neural network structures, networks CNN long short-term memory (LSTM) [34] were distinguished. Such a structure considers data in a specific environment and at a particular time interval, which is very suitable for recognizing human emotions.

In addition to the abovementioned types of neural networks, there are many others because new ideas and implementations appear every day in machine learning. A lot of those models are very complex but provide excellent accuracy. Unfortunately, complex systems are hard to use for real-time data analysis.

Multimodal data analysis for emotion recognition has made significant strides in recent years, but there are still several gaps and challenges that researchers are working to address. Some of these gaps include:

- While there has been progress in analyzing multiple modalities, integrating these modalities effectively remains a challenge. Combining information from different modalities in a coherent and complementary manner is essential for accurate emotion recognition.
- Emotion data often suffer from class imbalance, where certain emotions are underrepresented compared to others. Additionally, emotions are highly subjective and context-dependent, leading to variability in how they are expressed and perceived across different individuals and cultures.
- Many applications of emotion recognition, such as human–computer interaction and affective computing, require real-time processing of multimodal data. Achieving low-latency, real-time performance while maintaining high accuracy poses a significant technical challenge.

For real-data analysis with different modalities, it is worth accepting something in the middle, in order to implement faster online prediction on a client-server architecture. In particular, transformer structures that include an encoder and a decoder are well suited for that task. These systems are also very user friendly as they are often designed to learn to produce convenient tape output. It is this structure that we will explore further in the work. And also, LSTM is explored for sound and image modality.

In the paper, CNN + LSTM was chosen for our model, because this method is easy to understand and implement with an appropriated level of accuracy. Late fusion in various implementations was chosen as the method of multimodal data fusion. Number of emotions is one of the model's parameters. Bidirectional LSTM with CNN where the first layer is wrapped with a TimeDistributed function is used for face emotion recognition.

3. Materials and Methods

3.1. Data Preparation

The solution implemented in the paper is based on the combination of simple unimodal methods for emotion recognition into one multimodal method. This solution allows a reasonably accurate result to be obtained, for example, in recognizing a person's psychoemotional state while using the banking system [25]. Also, developing such an information system will make it possible to understand the advantages of multimodal recognition of emotions.

The most straightforward 3 options of the model and information system that can be implemented (Figure 1) are the following:

- Merger of three information streams at the stage when the probability is obtained for three models separately. In this case, we take a constant weighted sum of all three results (Figure 1a).
- Stream merging at the stage when each model has formed a certain set of characteristics for its data stream. In this case, we combine all these characteristics into one array and submit it for further processing with a typical output (Figure 1b).
- Data merging at the beginning of the neural network immediately after pre-processing (Figure 1c).

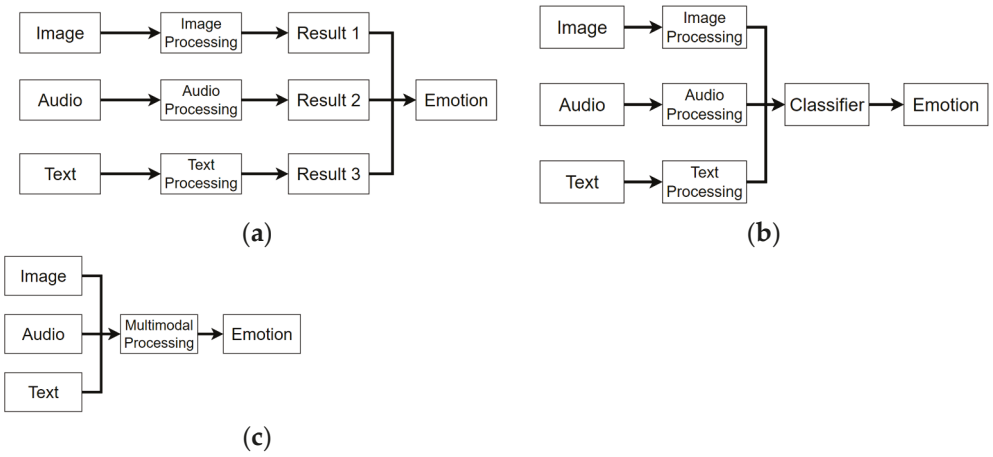


Figure 1. Diagrams of systems with different methods of merging (**a**) merging in the end; (**b**) merging in the middle; (**c**) merging in the start.

Multi-modal emotion recognition (MMER) system merges text, sound and images. The data flow in the system is realized not only from the camera and microphone, but also from the saved recording in the device memory and the screen broadcast. Based on the analyzed requirements, we will use CNN [33] and LSTM [34] models of neural networks for three types of data and different ways of merging them.

We can identify three points where new data arrives. The first moment is when we read data from the camera and microphone (or from a recording or broadcast of the screen) and submit it for processing. The second point is when the data is fed to the input of the neural network for the learning process. The third point is when pre-processed data is fed to a trained network to predict human emotions.

Based on characteristic of supervised learning in [35], we have to enter two data streams, x and y. In our case, y is the probability vector that the emotion belongs to a certain category. And x is either data of one type or an array of data of different types. This is also called the classification problem.

Consider the data for a text emotion recognition (TER) model, i.e., to recognize emotion in text. The data in these datasets are taken from the Kaggle resource [36]. There are about 5–15 words in one fragment of text. The total number of such fragments in the initial data set is 47,291.

Next, consider a similar folder for speech emotion recognition (SER). This is a RAVDESS dataset taken from the Kaggle resource [37]. Files are organized in folders based on different actors who voice emotions. The average recording time in these files is 3–6 min. The original data set consists of 1440 files.

Now let us look at a similar folder for training the neural network facial emotion recognition (FER). This is a data set CK + 48 with 5 emotions [38] taken from the resource Kaggle. Faces are manually cropped in that dataset [39]. Each folder has a sequence of 3 images, which is required for proper training of our LSTM model. Images contain the faces of different people. The size of the image is 48*48 in grayscale. The total number of images in this initial data set is 765. The Fer13 dataset [40] contains 8 emotions. This dataset can be used for testing the efficiency of the proposed model.

The amount of data in the sets can be easily increased due to the implemented functionality of data regeneration from other sets or from the live data stream from the camera and microphone (or from recording or broadcasting the screen). The preprocessing is used for that. At the output of the camera, we have an image in three RGB colors. We need to convert it to grayscale, find and cut out the human face, and reduce the image to 48*48 dimensions, as in the dataset we trained the neural measurer on.

In the end, unimodal datasets consist of:

- FER: 765 images from CK + 48 with surprise, sad, happy, fearful and angry expressions; calm emotion was added from 162 manually cropped images from live-stream (camera and YouTube (San Mateo, CA, USA)); in addition, the dataset was augmented using data augmentation generative adversarial network (DAGAN [41]) with 186 images;
- SER: 1440 files from RAVDESS with surprise, sad, happy, fearful, angry, calm and disgust expressions; disgust emotion files were eliminated;
- TER: 47,291 short texts with surprise, sad, happy, fearful, angry and calm emotions.

Unimodal datasets are used for unimodal neural network training. Then, the pretrained neural networks are used in merged neural networks. To train a multimodal system with neural networks, the functionality of processing and combining data of three types from these three data sets was implemented. In the first step, data from TER and SER were combined according to the same meaning of the text and the meaning of the emotion. In the second step, a sequence of 3 images with the same emotion value was added. This sequence is generated from 5–10 s video as 3 separated images with some difference between images. The dissimilarity is found using the package imagehash. The final dataset consists of 1167 FER + SEP + TER triplets and 162 triplets generated from live streaming and YouTube. The division into folders is implemented similarly to CK + 48 by emotions. An example of one package consisting of a text file, an audio recording and three images of an emotional face is shown in Figure 2.

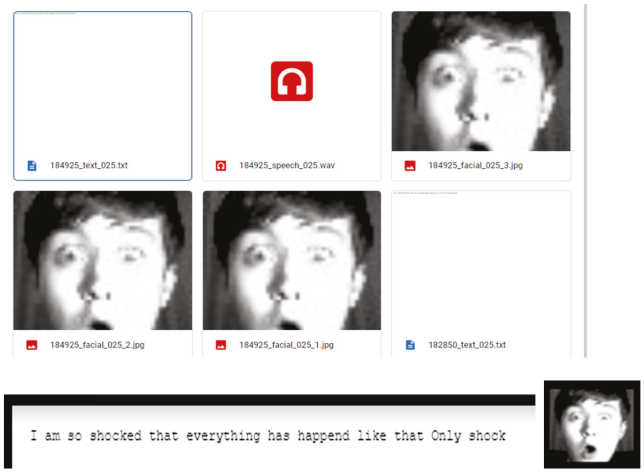


Figure 2. An example of one dataset for the emotion of surprise (https://doi.org/10.6084/m9.figshare.23596362.v1, accessed on 1 September 2022).

Of the dataset, 70% is used for training and 30% for testing.

It is possible to feed each type of data separately to its corresponding neural network. There is also a similar implementation in the case when this network is part of a large network with a merging in the middle (Figure 3a). Another presentation option is data fusion right at the beginning with image preprocessing for the adequacy of this fusion, and already a joint presentation at the entrance to the network (Figure 3b).

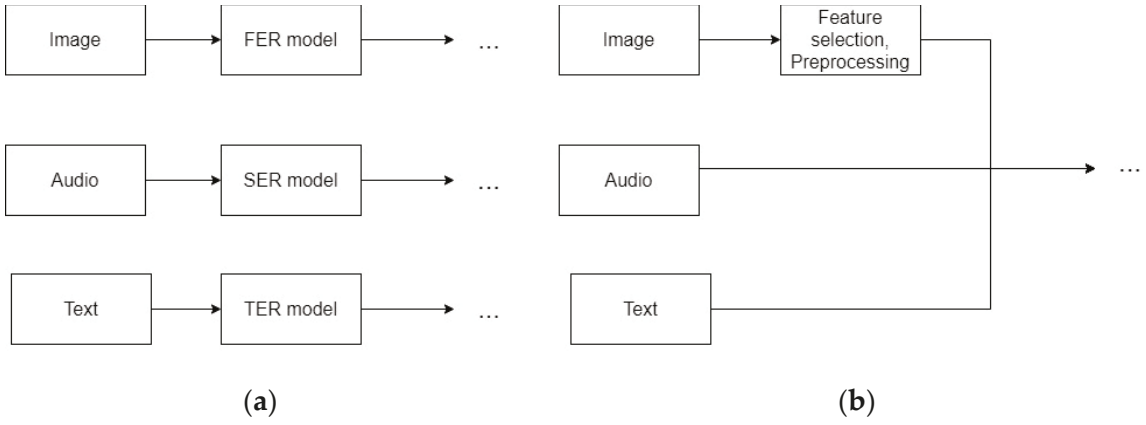


Figure 3. Input different types of data into their respective models: (**a**) to feed each type of data separately to its corresponding neural network; (**b**) data fusion right at the beginning with image preprocessing for the adequacy of this fusion.

3.2. Model Development

There are three cases of multimodal data analysis. In the first case, the multimodal emotion recognition model with late fusion is the structure organized as follows: the results of three modalities are obtained from the corresponding unimodal networks and then added with the same coefficient of influence on the final result (0.33). In the second realization, the multimodal emotion recognition model with middle fusion is the structure realized as follows: the characteristics of the three modalities obtained in the corresponding unimodal networks are added to the softmax layer and processed in several fully connected layers, which allows a well-balanced joint result to be obtained. The latter option, multimodal emotion recognition with early fusion, is organized as a structure where different types of data streams merge as early as possible and are processed as one common unimodality.

All previously mentioned options correspond to the late merger in the concept of methods of multimodal recognition of the psycho-emotional state, without modifications that take into account intermodal connections. As a result of the system, we expect to have a vector of probabilities that the emotion belongs to a certain category, or classification problem.

Once we have the data, we need to have neural networks. In our case, there are 5 neural networks for 3 systems for recognizing a psycho-emotional state. The implementation of those networks is presented below.

In the first system, we have 3 separate neural networks, each of which is responsible for its own type of input. But at the output, the prediction results of 3 neural networks merge into one.

The first network is FER—a neural network for facial emotion recognition. The architecture of this neural network is based on CNN and bidirectional LSTM (Figure 4) where the first layer is wrapped with a TimeDistributed function. The TimeDistributed layer requires a minimum of 3 dimensions. The objective is to consider the sequences of three images as a single input data because every sequence contains necessary information. Six emotions are taken into account.

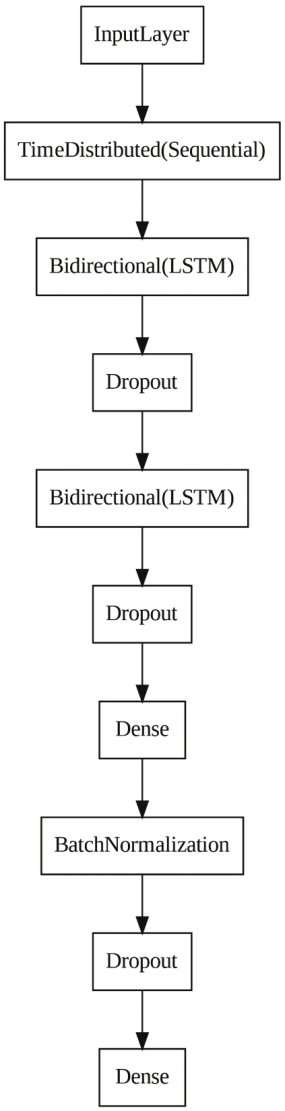


Figure 4. The structure of the FER neural network.

In this and all other neural networks, the size of the input and output layers of the network are parametric. This allows you to quickly change the number of emotions with which the neural network works.

The next speech emotion recognition (SER) network is a network for recognizing emotions by the sound a person makes when talking. This network has the simplest and small architecture. It allows you to quickly and effectively train and predict emotion based on the data received in the "live" mode. CNN is used here. The structure of the network is shown in Figure 5. Emotions are characterized by a different set of jumps in the spectrogram and different acoustic features.

Mel-spectrogram analysis is used for recognition of emotions. According to [42], a context window of 64 frames is adopted to extract the static Mel-spectrogram. The frame shift size of 32 frames is used to generate overlapping segments of the Mel-spectrogram. The overlapping segments is the key to speech segmentation. As a result, the static Mel-spectrogram is obtained at the size of 64*64. Therefore, 64 cores help to find such fluctuations.

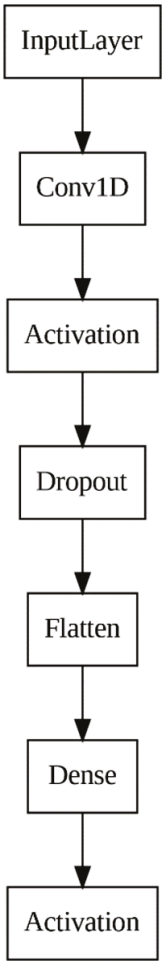


Figure 5. The structure of the SER neural network.

The last network in this system is TER—a network for recognizing emotion from text. This network has a very complex and bulky architecture. This is due to the general difficulty of determining emotions by text. CNN and LSTM are used here. The structure of this network is presented in Figure 6. A maxpooling layer is used after vertical concatenation.

The next system with a merger in the middle is implemented according to the principle of combining the 3 neural networks mentioned above with beige "softmax" layers (MMERm—multimodal emotion recognition with merge at the middle). That is, at the moment of obtaining certain characteristics. CNN LSTM is used here, accordingly. The structure of this neural network is cumbersome and is shown in Figure 7.

Merge at the end (MMERe—multimodal emotion recognition with merge at the end) is developed as the structure when the obtained results of three modalities, in the corresponding unimodal networks, are added with the same coefficient of influence on the final result (0.3 (3)). To do this, we use the previous models (FER, TER and SER).

The last version of the system is implemented according to the principle of merging the neural network immediately at the input (MMERs—multimodal emotion recognition with merge at start). A set of fully connected layers is used here. The idea of such a structure was to merge the data into one row at the very first step and highlight hidden dependencies and characteristics with the help of several fully connected layers. First of all, this network will be used as an experiment with various structures that can only be invented. The structure of this neural network is shown in Figure 8.

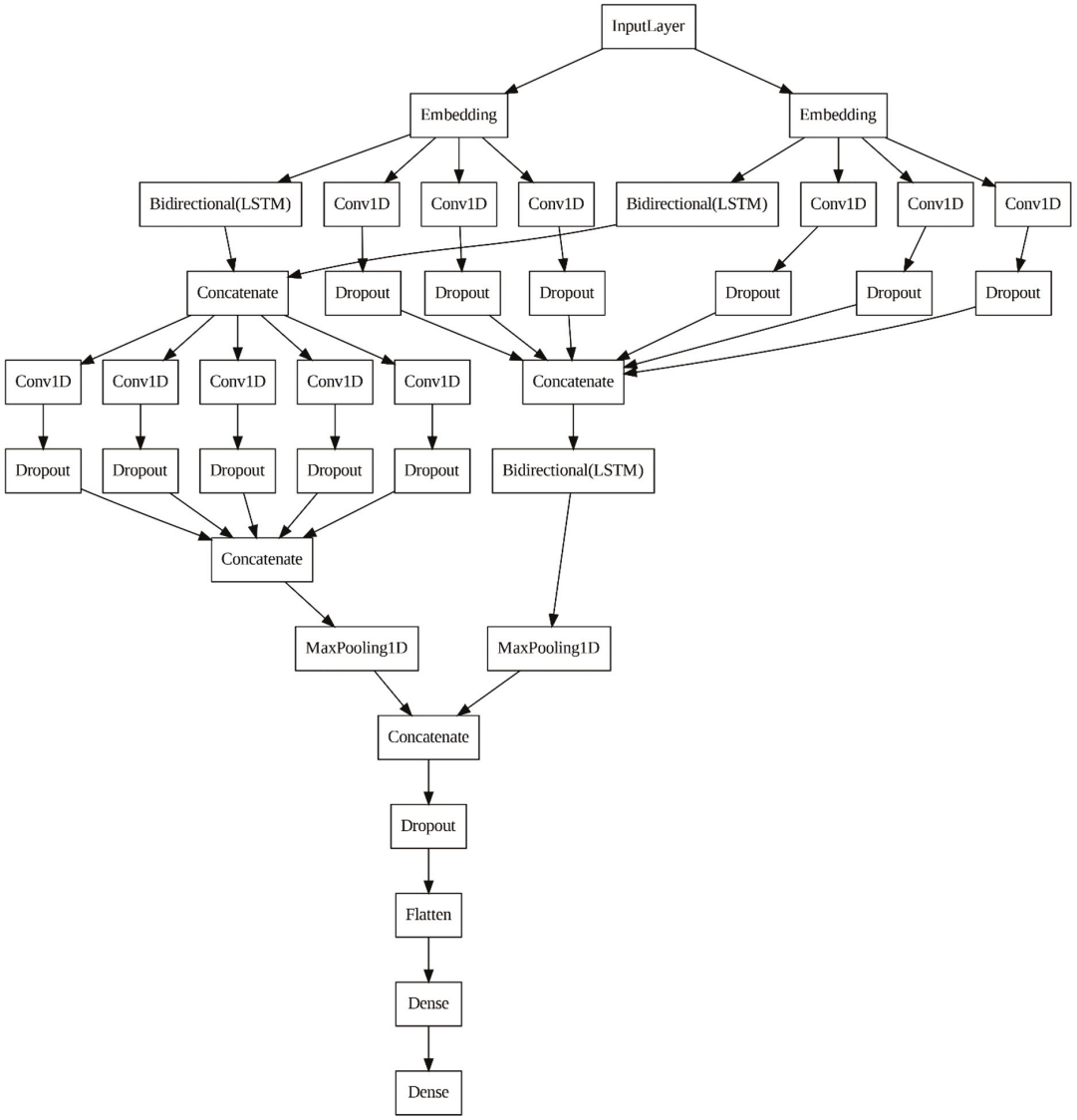


Figure 6. The structure of the TER neural network.

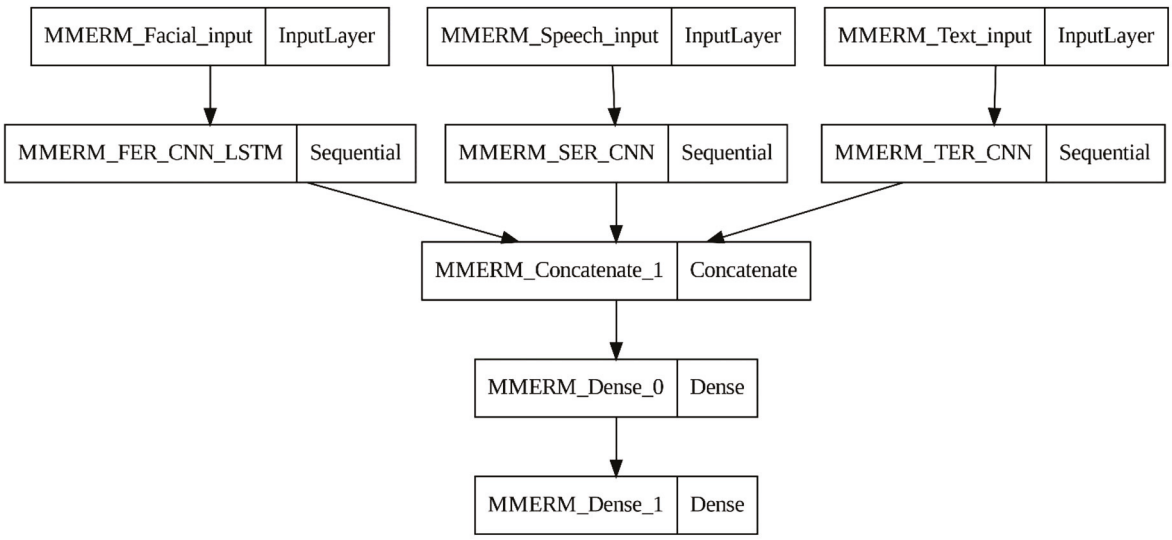


Figure 7. The structure of the MMERm neural network.

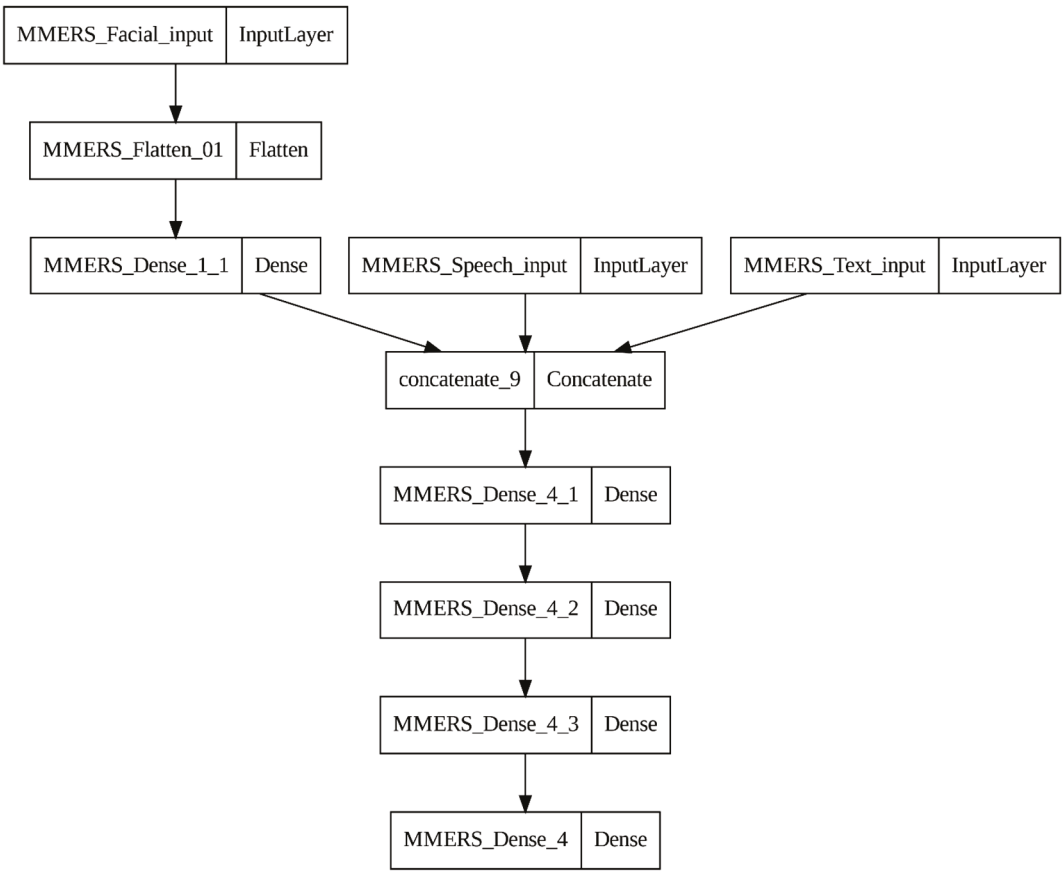


Figure 8. The structure of the MMERs neural network.

4. Results

The system was developed based on the following technologies algorithms and frameworks:

- CNN, LSTM
- Dropout, MaxPooling, Flatten, Dense, Reshape, GlobalMaxPool2D, BatchNormalization, Bidirectional, TimeDistributed, Embedding;
- Nadam;
- Relu, softmax, elu;
- EarlyStopping, ReduceLROnPlateau;
- Accuracy, categorical_crossentropy;
- Precision, recall, f1-score, support.

In addition:

- PCA, TSNE;
- OpenCV 4.8.1 (convertScaleAbs, LUT);
- Ffmpeg 0.2.0 -python (ffmpeg.input);
- Librosa 0.10.1 (feature.mfcc);
- SpeechRecognition (speech_recognition.recognize_google);
- Tokenizer (texts to sequences).

For results evaluation, the following metrics are used: accuracy, categorical_crossentropy, precision, recall, f1-score.

Accuracy is the percentage of correctly classified data (*tp*—true positive, *tn*—true negative, *fp*—false positive, *fn*—false negative):

$$A = \frac{tp + tn}{tp + tn + fp + fn}.$$

Categorical cross-entropy is a loss function that calculates the difference between two probability distributions: the observed distribution and the predicted probability distribution. In the case of classification, the input is fed to the input of the model, which generates a vector of distribution probabilities across all classes. The next step is to compare this vector with the vector of actual values and calculate the cross-entropy between the two vectors.

The basic idea behind categorical cross-entropy is that we penalize a model if its predicted probabilities differ from the actual values. Therefore, the goal of optimization is to reduce this loss by adjusting the model parameters.

Because neural networks are used not only in the learning process, where logits immediately hit the evaluation function but also to show the percentage of predicted emotion, the result normalized by the Softmax layer is more presentable and understandable. If we consider the context of the study of structures on metrics, the softmax layer can be omitted and specify the CrossEntropyloss function with the from_logits = true parameter, in which case the internal code would do the same as the softmax layer. Therefore, this decision has no effect on the result. The separate layer was made for convenience in our context.

Precision is the proportion of relevant instances among the extracted instances:

$$P = \frac{tp}{tp + fp}.$$

Recall is the proportion of relevant instances that were retrieved:

$$R = \frac{tp}{tp + fn}.$$

$F1$-score combines precision and recall and takes their harmonic mean. It is used to compare the effectiveness of classifications:

$$F1 = \frac{2 \times (P \times R)}{P + R}.$$

Python 3 Google Compute Engine backend (GPU) was used for experiments performing.

To begin, let us look at detailed results for training and testing using six emotional states.

For qualitative assessment, TSNE derivation is also implemented for the characteristics formed in each network at the penultimate layer of the neural network during prediction. The program uses “PCA” and “TSNE” implementations for this. For example, on the validation data for five neural networks, we will have the graphs shown in Figures 9 and 10. Therefore, the cluster density for FER is the best compared with the rest of the unimodal models. The SER model demonstrates clusters overlapping. MMERm and MMERs demonstrate appropriate cluster density too.

And finally, we visualize the TSNE graphs for three architectures for test dataset (Figures 11 and 12). TSNE converts similarities between data points to joint probabilities and tries to minimize the Kullback–Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data (based on scikit-learn version 0.16.1 documentation [43]). The obtained result is the same. The best result is for MMERm, MMERs and FER. SER presents clusters overlapping. Anger emotion is well recognized based on MMERm.

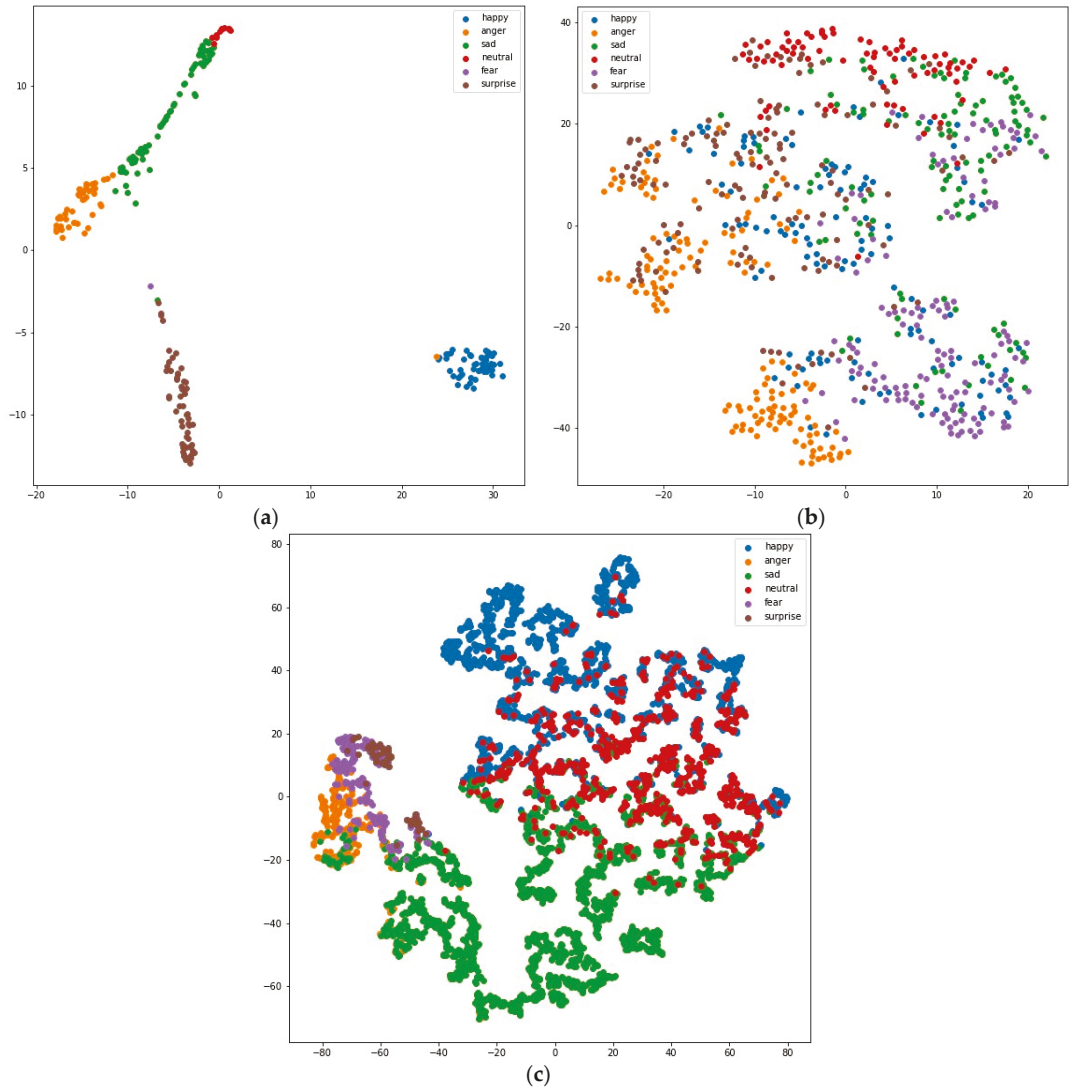


Figure 9. TSNE for FER (**a**), SER (**b**), TER (**c**)—training dataset.

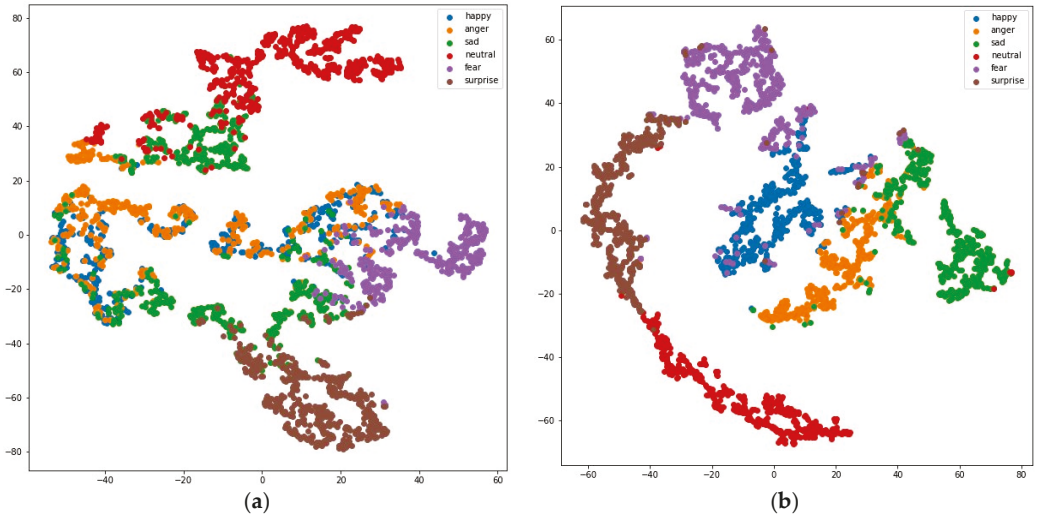


Figure 10. TSNE for MMERm (**a**) and MMERs (**b**)—training dataset.

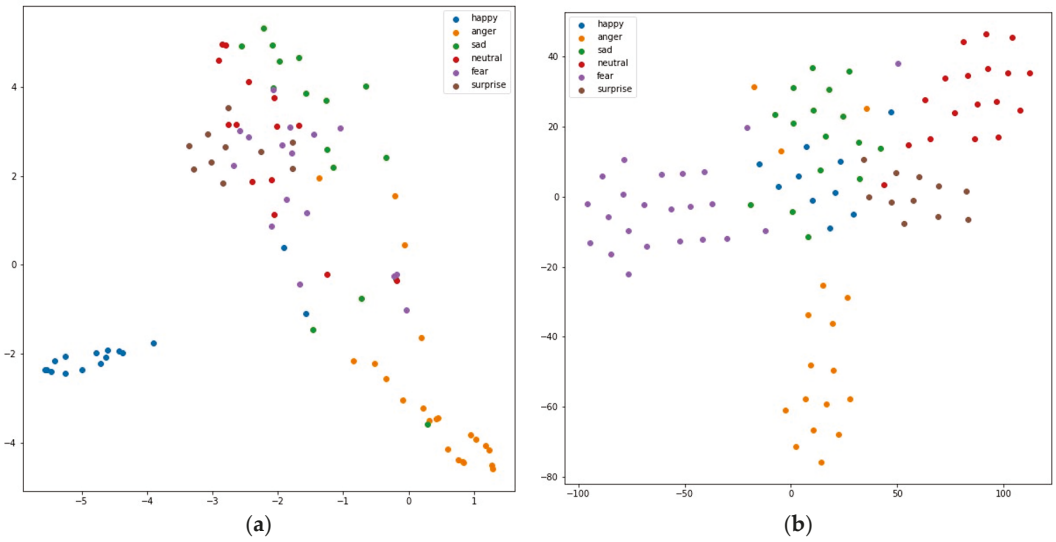


Figure 11. TSNE for MMERs (**a**), MMERm (**b**)—testing dataset.

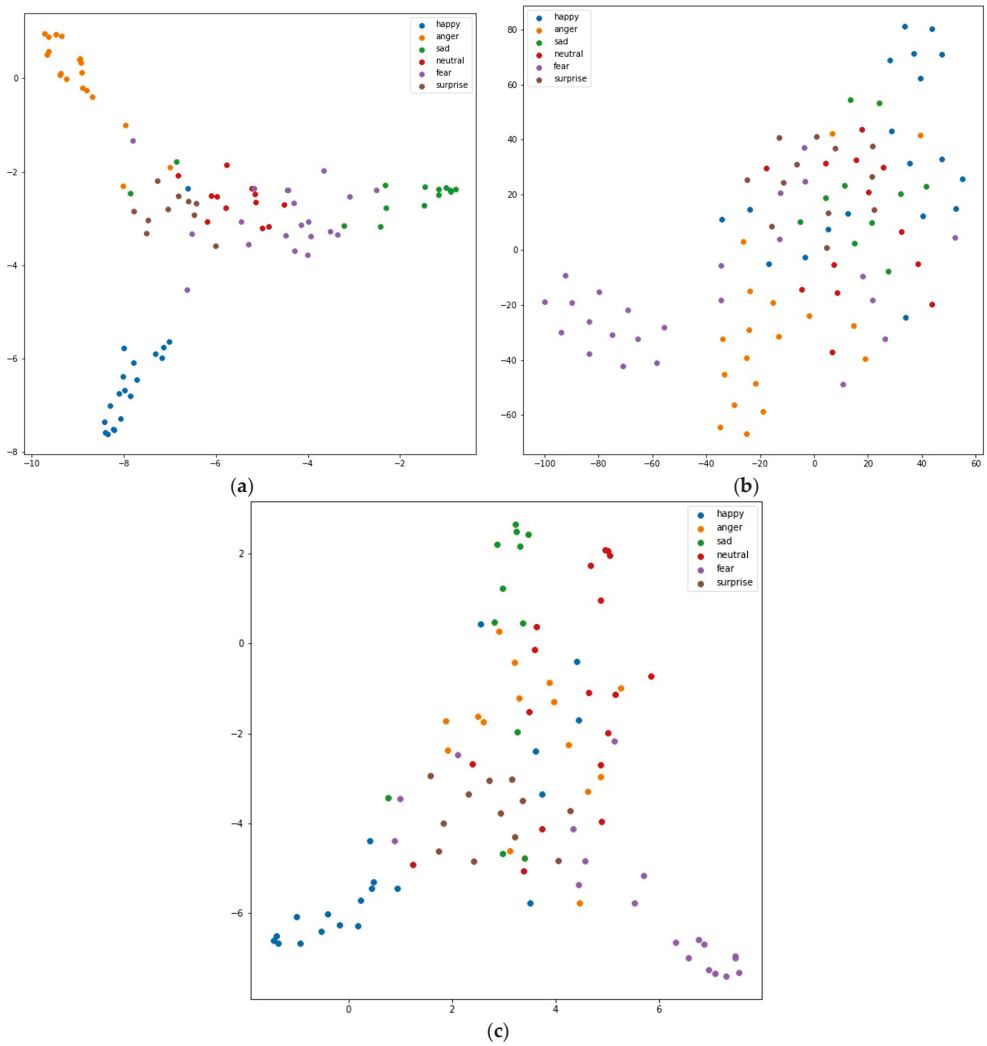


Figure 12. TSNE for FER (**a**), SER (**b**), TER (**c**)—testing dataset.

Based on these three conclusions, we are able to qualitatively analyze the results of testing in "live" mode.

First, Table 1 shows the resulting accuracy of five neural networks. The highest validation accuracy based on multimodal data is obtained for MMERS. However, multimodal accuracy is still worse than the accuracy for unimodal FER. FER, SER and TER were trained on a unimodal separated dataset, then the pretrained neural networks are used in merged neural networks.

Table 1. Table of neural network training results.

	MMERe			**MMERm**	**MMERs**
	FER	**SER**	**TER**		
val_accuracy	0.9245	0.4363	0.6416	0.7000	0.7347
val_loss	0.4402	0.9542	0.7048	0.4647	0.6396
train_accuracy	1.0000	0.6544	0.8209	0.9583	0.9774
train_loss	0.0267	0.8468	0.4675	0.0732	0.05832

Table 2 shows the metrics of testing on validation data. Metric values will be presented as "weighted avg". However, metrics "micro avg", "macro avg" and "samples avg" are also available for analysis. The best result is obtained for FER (video modality) and the worst result is obtained for SER (audio modality). The difference in accuracy across modalities is explained by the difference in the quality of the datasets. So, for images, a sequence of three images is analyzed. A convolutional neural network works well for recognizing individual images. Moreover, accuracy increases with combined results of CNN and LSTM for sequential images. Speech recognition is performed for a dataset consisting of audio files 4 s long, due to which the end of the sentence to determine the emotion is not always correct. The same situation exists for the text dataset. At the same time, the combination of three modalities makes it possible to significantly improve the result.

Table 2. Table of test results of networks after training.

	MMERe			**MMERm**	**MMERs**
	FER	**SER**	**TER**		
precision	0.92	0.56	0.64	0.92	0.90
recall	0.92	0.56	0.64	0.90	0.89
f1-score	0.92	0.56	0.64	0.91	0.89
support	106	319	16377	120	720

For a better understanding, we display this data in the form of a bar chart in Figure 13.

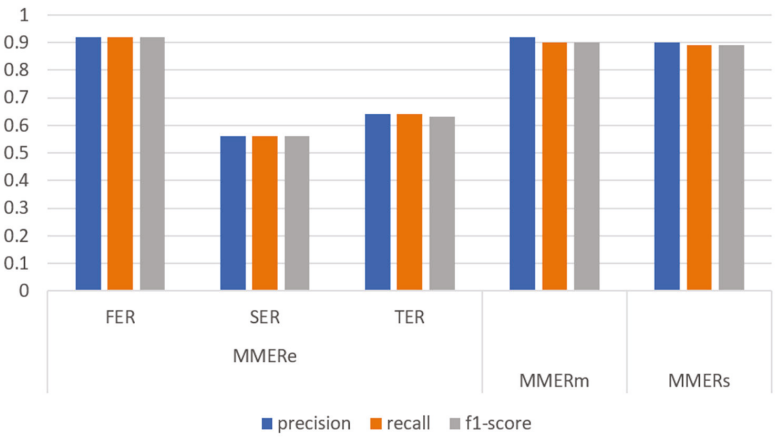
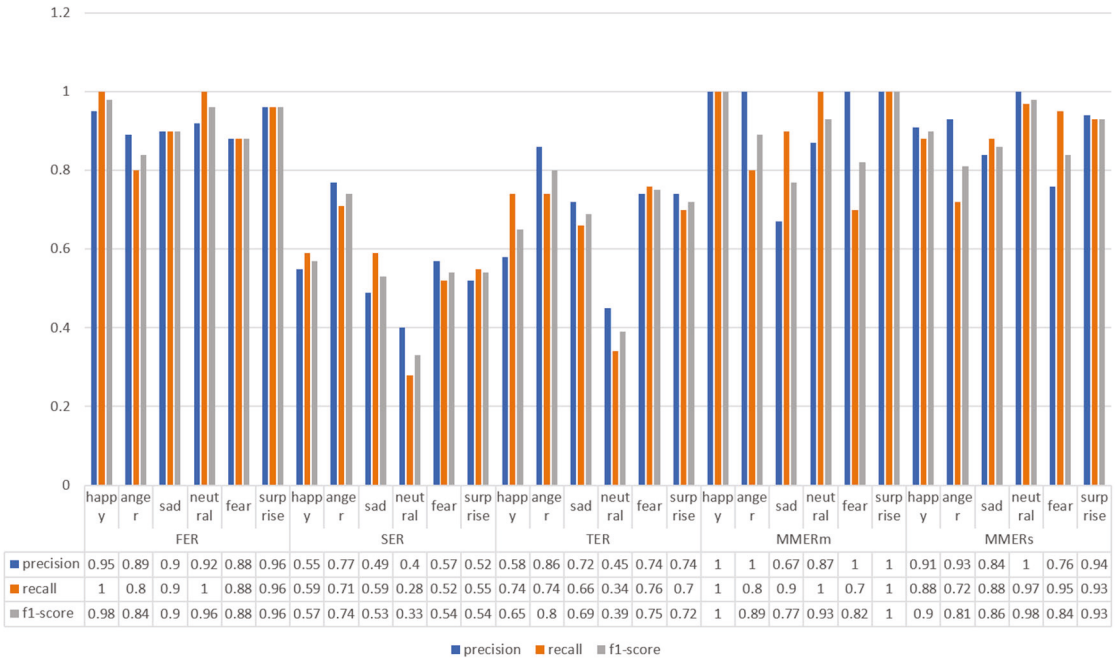


Figure 13. Bar chart of average training results.

The value of such metrics can be obtained for each emotion separately.
For a better understanding, we display this data in the form of a bar chart in Figure 14.



	FER						SER						TER						MMERm						MMERs					
	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise
precision	0.95	0.89	0.9	0.92	0.88	0.96	0.55	0.77	0.49	0.4	0.57	0.52	0.58	0.86	0.72	0.45	0.74	0.74	1	1	0.67	0.87	1	1	0.91	0.93	0.84	1	0.76	0.94
recall	1	0.8	0.9	1	0.88	0.96	0.59	0.71	0.59	0.28	0.52	0.55	0.74	0.74	0.66	0.34	0.76	0.7	1	0.8	0.9	1	0.7	1	0.88	0.72	0.88	0.97	0.95	0.93
f1-score	0.98	0.84	0.9	0.96	0.88	0.96	0.57	0.74	0.53	0.33	0.54	0.54	0.65	0.8	0.69	0.39	0.75	0.72	1	0.89	0.77	0.93	0.82	1	0.9	0.81	0.86	0.98	0.84	0.93

Figure 14. Bar chart of training results for each emotion.

In addition to validation training, it is important to analyze live test results, as these are important in our system of multimodal recognition of a person's psycho-emotional state in the data stream from the camera and microphone or recording or broadcasting the user's screen.

First, we present the results of various metrics with averaging "weighted avg" (Table 3).

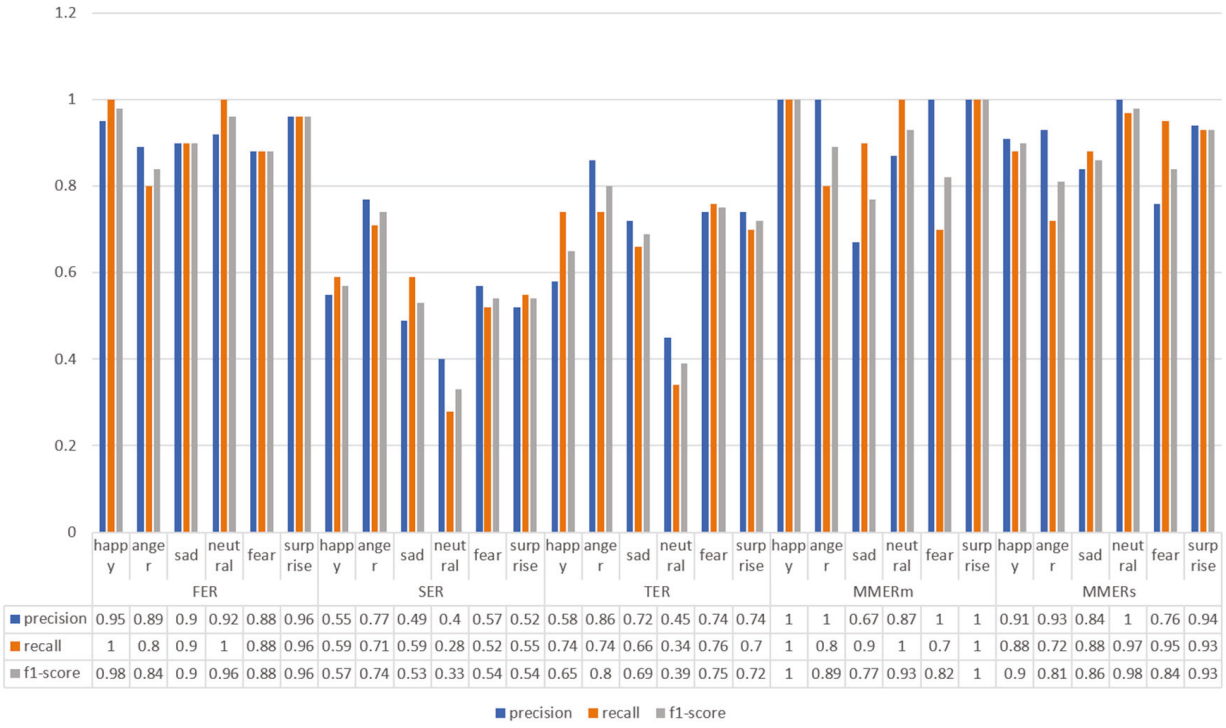
Table 3. Table of test results of networks after testing.

	MMERe			**MMERm**	**MMERs**
	FER	**SER**	**TER**		
precision	0.78	0.54	0.67	0.71	0.62
recall	0.66	0.48	0.63	0.62	0.58
f1-score	0.715	0.51	0.65	0.66	0.60
support	67	31	29	62	52

The results of the same metrics for each emotion separately are given in Figure 15. Happy emotion is the best recognized by FER, SER, MMERm and MMERs. Neutral emotion is the hardest emotion for recognition by SER and TER.

We will also show an alternative version of the visualization of test results for each emotion in Figure 16.

Next, the training time for different numbers of emotions and accuracy were analyzed. All neural networks were trained on the same dataset. Figure 17 presents time dependence of number of emotions. The maximum number of emotions for multimodal data is six because the same number of emotions is used in the FER dataset.



	FER						SER						TER						MMERm						MMERs					
	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise	happy	anger	sad	neutral	fear	surprise
precision	0.95	0.89	0.9	0.92	0.88	0.96	0.55	0.77	0.49	0.4	0.57	0.52	0.58	0.86	0.72	0.45	0.74	0.74	1	1	0.67	0.87	1	1	0.91	0.93	0.84	1	0.76	0.94
recall	1	0.8	0.9	1	0.88	0.96	0.59	0.71	0.59	0.28	0.52	0.55	0.74	0.74	0.66	0.34	0.76	0.7	1	0.8	0.9	1	0.7	1	0.88	0.72	0.88	0.97	0.95	0.93
f1-score	0.98	0.84	0.9	0.96	0.88	0.96	0.57	0.74	0.53	0.33	0.54	0.54	0.65	0.8	0.69	0.39	0.75	0.72	1	0.89	0.77	0.93	0.82	1	0.9	0.81	0.86	0.98	0.84	0.93

Figure 15. Bar chart of test results for each emotion.

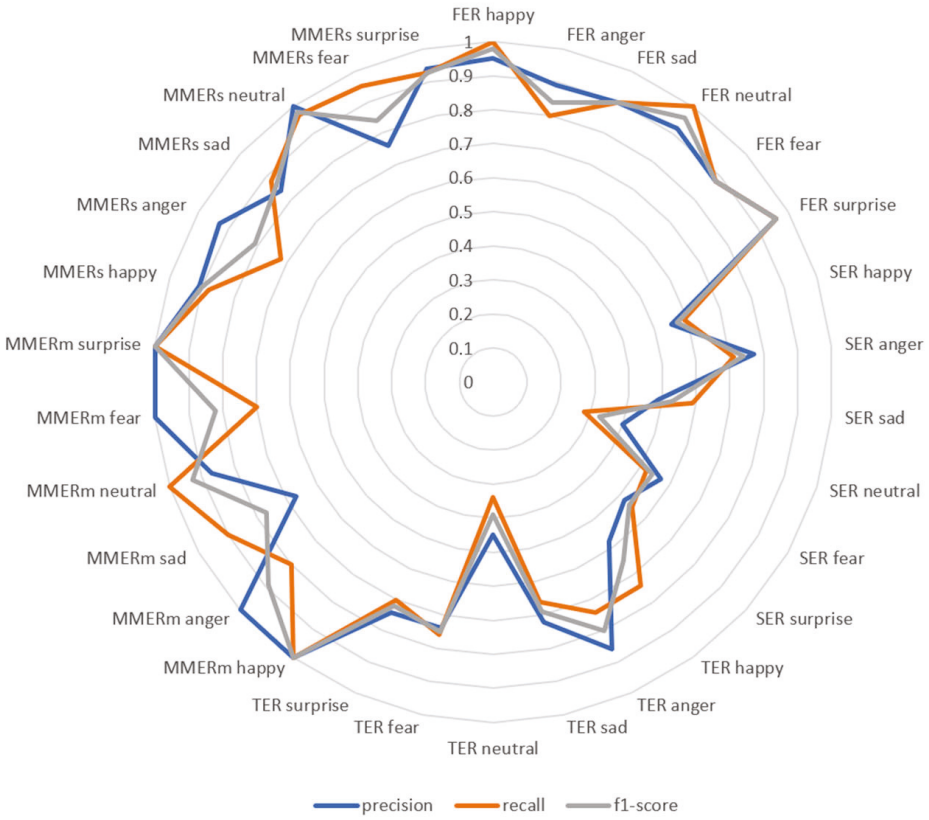


Figure 16. Drawing test results for each emotion.

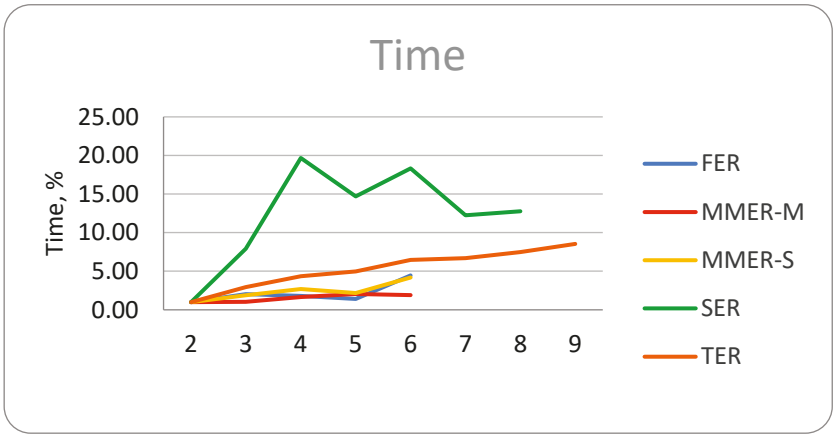


Figure 17. Training time dependence from number of emotions.

More detailed information is presented in Tables 4 and 5.

Table 4. Accuracy and training time for different numbers of emotions (from nine to six emotions).

Model	Number of Emotions											
	9			8			7			6		
	Time (s)	Accuracy (train/val)	Epochs	Time (s)	Accuracy (train/val)	Epochs	Time (s)	Accuracy (train/val)	Epochs	Time (s)	Accuracy (train/val)	Epochs
FER	-	-	-	-	-	-	-	-	-	310.12	1.0/0.92	464
SER	-	-	-	36	0.67/0.47	138	34.74	0.71/0.55	141	52.03	0.72/0.55	250
TER	725.43	0.75/0.66	48	634	0.75/0.65	45	567.93	0.74/0.66	45	547.73	0.74/0.65	45
MMER-S	-	-	-	-	-	-	-	-	-	29.51	1.00/0.97	78
MMER-M	-	-	-	-	-	-	-	-	-	195.6	0.99/0.85	18

Table 5. Accuracy and training time for different numbers of emotions (five to two emotions).

Model	Number of Emotions											
	5			4			3			2		
	Time (s)	Accuracy (train/val)	Epochs	Time (s)	Accuracy (train/val)	Epochs	Time (s)	Accuracy (train/val)	Epochs	Time (s)	Accuracy (train/val)	Epochs
FER	96.8	0.87/0.86	157	125.61	0.99/0.88	222	140.25	1.0/0.85	347	69.57	1.0/1.0	196
SER	41.71	0.73/0.64	235	55.85	0.77/0.66	346	22.45	0.80/0.75	176	2.84	0.59/0.75	21
TER	421.23	0.74/0.65	36	369.32	0.86/0.78	38	249.9	0.97/0.94	40	84.88	0.98/0.95	66
MMERs	15.28	0.96/0.90	43	18.94	1.00/0.94	71	13.32	1.00/0.89	57	7.062	1.00/0.94	39
MMERm	211.58	0.98/0.82	22	170.03	0.99/0.95	24	105.82	1.00/0.95	16	102.76	0.99/1.00	26

The list of emotions is presented below:

9 “neutral”, “happy”, “sad”, “hate”, “anger”, “joy”, “fear”, “love”, “surprise”
8 “neutral”, “happy”, “sad”, “anger”, “joy”, “fear”, “love”, “surprise”
7 “neutral”, “happy”, “sad”, “anger”, “fear”, “love”, “surprise”
6 “neutral”, “happy”, “sad”, “anger”, “fear”, “surprise”
5 “neutral”, “happy”, “sad”, “anger”, “surprise”
4 “happy”, “sad”, “anger”, “surprise”
3 “sad”, “anger”, “surprise”
2 “anger”, “surprise”

From Tables 4 and 5, we can make a conclusion about the dependence of learning time on the number of emotions. Also, the accuracy of recognition slightly decreases with the increase in the number of emotions. Although for six emotions, the accuracy is higher than for five. This situation is explained by the features of the data sets and the number of emotions that were basically presented in them.

The difference in train and validation accuracy for different numbers of emotions can be explained in the following way. Some emotions are similar and that is why it is very hard to recognize them based on only one modality. From the other side, MMERs looks the best in comparison with the rest of the models.

The next experiment was conducted based on size of images. The purpose of the experiment was to test the hypothesis of whether a larger image size would affect the accuracy of emotion classification. To conduct the experiment, two options for changing the data were used. The first option was to artificially increase the image size of the original CK + 48 dataset to 192*192. The second way was to use new data obtained from a real data stream (noted with symbol +) in which the frame size was 192*192. Also, for comparison, the new images were reduced to 48*48. Accordingly, for the analysis of new data, the FER architecture in Figure 4 has also been resized to fit the image. The change in the training time of the FER neural network depending on the image size was also analyzed.

Early stopping is used to increase the efficiency of training neural networks and to avoid overfitting. On other hand, one of the drawbacks of early stopping is that it can be sensitive to the choice of the validation set, the criterion, and the threshold or the patience parameter. To analyze the quality of developed FER NN, three early stopping criteria were implemented based on:

- Validation loss,
- Number of epochs,
- Accuracy.

The first experiment was conducted based on validation loss as the early stopping criteria. Figures 18 and 19 represent validation accuracy and training time, respectively, for different numbers of emotions. The comparison in Figure 18 is presented on an absolute scale. The time percentage in Figure 19 expresses the training time increasing when comparing two emotions. In this figure, the relative time for two emotions is equal to one, the relative time of the rest of the emotions is presented as relative time increasing.

In Figure 19, for three emotions and for six emotions training time is higher for smaller images (48*48) than for bigger images (192*192). One possible explanation for that is lower recognition rate for smaller images. The classification accuracy (Figure 18) on the new 192*192 images is slightly lower than on the original SK + 48 dataset. Obviously, for artificially reduced or artificially enlarged images, the classification accuracy is much lower. The analysis of computational complexity (Figure 19) showed the dependence of time on the number of emotions and the size of images.

The number of epochs was chosen as a second experimental stopping criteria. Due to limited computation resources for multiple experiments, the number of epochs is equal to 20. The validation accuracy and validation loss are given in Figure 20. Based on the presented results, the decision is that the number of epochs is not sufficient to train the models, especially for FER (192*192).

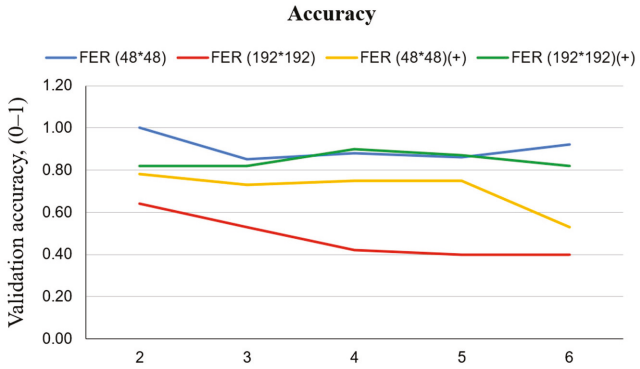


Figure 18. Accuracy dependence from number of emotions and image size for FER.

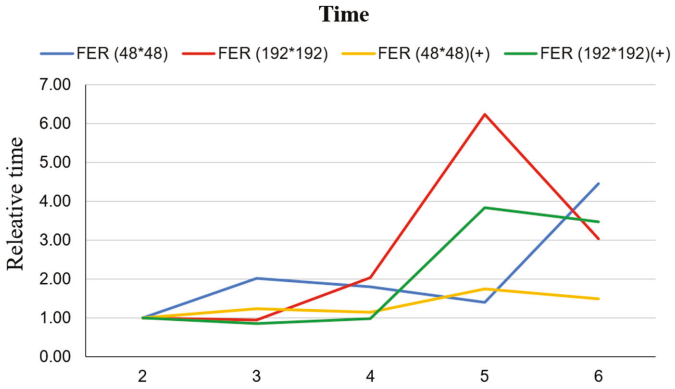
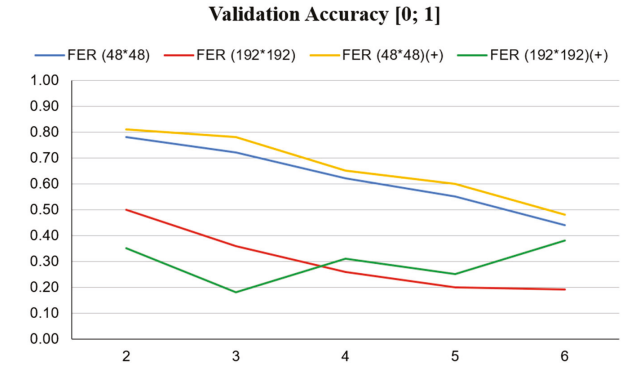
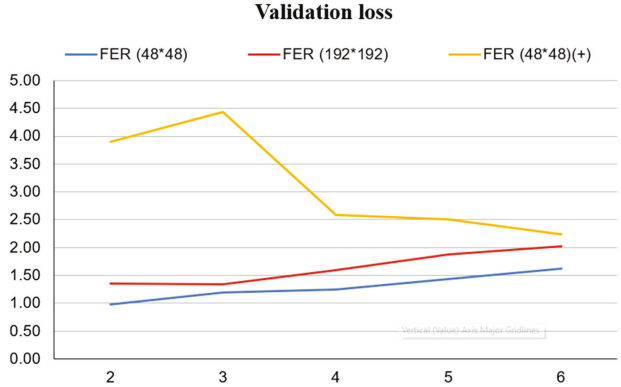


Figure 19. Training time dependence from number of emotions and image size for FER.



(**a**)



(**b**)

Figure 20. Validation accuracy (**a**) and loss (**b**) for 20 epochs.

The next stopping criteria is built on accuracy. The value 0.4 is chosen for experimental purposes only and used for time evaluation and evaluation of the number of epochs (Table 6). Thus, the increasing number of epochs and training time depending on number of emotions and image size is presented.

Table 6. Training time and number of epochs to obtain accuracy equal to 0.4 or high.

Number of Emotions/ Model	FER (48*48)		FER (192*192)	
	Training Time	Number of Epochs	Training Time	Number of Epochs
6	9.55	33	14.84	26
5	8.83	28	8.3	12
4	6.33	13	6.03	6
3	4.62	2	4.66	2
2	4.42	1	4.48	2

Based on experiments presented in Figures 18–20, the best accuracy is obtained using early stopping criteria on loss. The training time depends on number of emotions, size of images and number of epochs.

The next experiment was a study of the dependence of classification accuracy on the number of images in the sequence (sequence length of the frames was used as input of the LSTM) used in FER. As can be seen from Table 7, increasing the number of images does not significantly affect the classification accuracy.

Table 7. Accuracy for different numbers of emotions and numbers of images in the sequence.

Emotion Number/ Validation Accuracy (%)	FER (48*48) + 3 Images	FER (48*48) + 5 Images
6	0.920	0.915
5	0.864	0.865
4	0.882	0.893
3	0.854	0.863
2	1.000	1.000

5. Discussion

From the results shown in Tables 1–3 and Figures 13–17, the following conclusions can be drawn.

Among the individual neural networks of the first MMERe system, the best and fairly good accuracy in the FER network is 0.9245. However, due to the low accuracy of SER and TER—0.5611 and 0.64, this system will show not so good quality in recognition based on multimodal data.

The other two options for merging data, MMERm and MMERs, trained better. The average accuracy of the models is 0.90 and 0.89, respectively.

Number of emotions is one of the model's parameters and can be easily changed.

However, there is a high probability that the networks in the last two systems have been retrained. This is confirmed by the data we received after testing in Table 3. In addition to the problems in the testing process mentioned above, one we must also highlight is the possible impact of fuzzy pronunciation on records, poor lighting, and other details that affect data quality.

The future research will be connected with MOSEI [44] dataset usage and comparison with existing results. The next step will be to find additional computing resources for training networks with the ability to work with images with a higher resolution than 48*48.

6. Conclusions

This work demonstrates the use of various simple methods of late fusion of multimodal data to achieve the best recognition accuracy. Based on the analysis, it was determined that the best among the proposed systems is the system with data fusion after processing each channel separately and obtaining individual characteristics. As the conclusion, we would like to note that marking the data obtained in live testing is a very long and difficult process. First, there is the subjective understanding of what the emotion is, which may not coincide with reality due to the similarity of the manifestations of psycho-emotional states and specifics depending on the author in the video and the circumstances. Secondly, for the amount of data that will allow adequately assessing the results, it is necessary to carry out an extremely long process of marking the data, which must be taken from a very long record of emotional manifestations of different people. Third, as predicted, on live data the accuracy will be much lower than on an almost identical, well-processed data set, which trains the neural networks of multimodal emotion recognition systems.

Therefore, for a correct analysis, it is necessary to conduct a much larger study, after which the correct conclusion can be determined.

The topic covered in this paper is very extensive. It takes a lot of time to cover at least part of this area. This topic is attractive, with its diversity and the results of the work of systems that use the technology of multimodal recognition of emotions based on machine learning.

According to [45], the highest recognition accuracy of 99% for FER is provided by the SVM classifier and it recognizes seven emotions. As obtained in the paper, FER's 92% accuracy is lower. However, the proposed model is tested for "live" video too with 79% accuracy, and that is impossible for SVM. In 2D FER, mostly JAFFE and CK databases are used for more efficient performance than the other databases.

When it comes to results presented in Tables 4 and 5, multimodal neural networks, particularly MMERs, present the best result for all numbers of emotions. It can be explained by training of all networks on the same dataset and increasing the accuracy based on merging streams.

Author Contributions: Conceptualization, N.S. and S.L.; methodology, O.Z.; software, O.Z.; validation, N.S., O.Z. and S.L.; formal analysis, O.Z.; investigation, N.S.; resources, S.L.; data curation, O.Z.; writing—original draft preparation, O.Z.; writing—review and editing, N.S.; visualization, O.Z.; supervision, N.S.; project administration, S.L.; funding acquisition, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: The National Research Foundation of Ukraine funded this research under project number 2021.01/0103 and British academy fellowship number RaR\100727.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting this study's findings are openly available in https://doi.org/10.6084/m9.figshare.23596362.v1 (accessed on 1 September 2022). Datasets are used for different models' performance evaluation, namely: FER: fer2013: https://www.kaggle.com/deadskull7/fer2013 (accessed on 1 September 2022) and CK + 48 five emotions: https://www.kaggle.com/gauravsharma99/ck48-5-emotions (accessed on 1 September 2022); SER: RAVDESS Emotional speech audio: https://www.kaggle.com/uwrfkaggler/ravdess-emotional-speech-audio (accessed on 1 September 2022); TER: Text-Emotion-detection: https://www.kaggle.com/dataset/f10c38f8f356a43b344ca82476b6b32b5d31b99af19276ba1f7846004c0851f2 (accessed on 1 September 2022); Datasets from the Internet inside the project: https://drive.google.com/drive/folders/1ZV3ceCjNND7xcUxbsJb57aitTpUbcYa9?usp=sharing (accessed on 1 September 2022); Videos for tests from YouTube: (1) Biden Delivers Remarks On Inflation_NBC News—https://www.youtube.com/watch?v=ckCOF719atE (accessed on 1 September 2022); (2) Boris Johnson_Ukraine will win war and 'be free'—https://www.youtube.com/watch?v=WPM8Pvgkz7Y (accessed on 1 September 2022); (3) Father's final words to his dying son!—https://www.youtube.com/watch?v=C3hABRHmQoo (accessed on 1 September 2022); (4) Minecraft Warden Update is a NIGHTMARE!—https://www.

youtube.com/watch?v=2osdz9Z5JKY (accessed on 1 September 2022). Video for Live Test: https://drive.google.com/drive/folders/1wAR2CdlGIEtOSjKv7T9e-gQhBHIAiLLM?usp=sharing (accessed on 1 September 2022).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Jena, P.K.; Ghosh, S.; Koley, E.; Mohanta, D.K.; Kamwa, I. Design of AC state estimation based cyber-physical attack for disrupting electricity market operation under limited sensor information. *Electr. Power Syst. Res.* **2022**, *205*, 107732. [CrossRef]
2. Qin, B.; Liu, D.; Chen, G. Formal modeling and analysis of cyber-physical cross-space attacks in power grid. *Int. J. Electr. Power Energy Syst.* **2022**, *141*, 107790. [CrossRef]
3. Cao, G.; Gu, W.; Lou, G.; Sheng, W.; Liu, K. Distributed synchronous detection for false data injection attack in cyber-physical microgrids. *Int. J. Electr. Power Energy Syst.* **2022**, *137*, 107788. [CrossRef]
4. Tahoun, A.H.; Arafa, M. Secure control design for nonlinear cyber–physical systems under DoS, replay, and deception cyber-attacks with multiple transmission channels. *ISA Trans.* **2021**, *128*, 294–308. [CrossRef]
5. Stellios, I.; Kotzanikolaou, P.; Grigoriadis, C. Assessing IoT enabled cyber-physical attack paths against critical systems. *Comput. Secur.* **2021**, *107*, 102316. [CrossRef]
6. Jena, P.K.; Ghosh, S.; Koley, E. Design of a coordinated cyber-physical attack in IoT based smart grid under limited intruder accessibility. *Int. J. Crit. Infrastruct. Prot.* **2021**, *35*, 100484. [CrossRef]
7. Li, L.; Wang, W.; Ma, Q.; Pan, K.; Liu, X.; Lin, L.; Li, J. Cyber attack estimation and detection for cyber-physical power systems. *Appl. Math. Comput.* **2021**, *400*, 126056. [CrossRef]
8. Ding, D.; Han, Q.-L.; Xiang, Y.; Ge, X.; Zhang, X.-M. A survey on security control and attack detection for industrial cyber-physical systems. *Neurocomputing* **2018**, *275*, 1674–1683. [CrossRef]
9. Lima, P.M.; Carvalho, L.K.; Moreira, M.V. Detectable and Undetectable Network Attack Security of Cyber-physical Systems. *IFAC-Pap.* **2018**, *51*, 179–185. [CrossRef]
10. Barrère, M.; Hankin, C.; Nicolaou, N.; Eliades, D.G.; Parisini, T. Measuring cyber-physical security in industrial control systems via minimum-effort attack strategies. *J. Inf. Secur. Appl.* **2020**, *52*, 102471. [CrossRef]
11. Liu, Y.; Deng, L.; Gao, N.; Sun, X. A Reliability Assessment Method of Cyber Physical Distribution System. *Energy Procedia* **2019**, *158*, 2915–2921. [CrossRef]
12. Qi, Q.; Lin, L.; Zhang, R. Feature Extraction Network with Attention Mechanism for Data Enhancement and Recombination Fusion for Multimodal Sentiment Analysis. *Information* **2023**, *12*, 342. [CrossRef]
13. Naseem, M.T.; Seo, H.; Kim, N.H.; Lee, C.S. Pathological Gait Classification Using Early and Late Fusion of Foot Pressure and Skeleton Data. *Appl. Sci.* **2024**, *14*, 558. [CrossRef]
14. Hu, A.; Seth, F. Multimodal sentiment analysis to explore the structure of emotions. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 350–358.
15. Williams, J.; Kleinegesse, S.; Comanescu, R.; Radu, O. Recognizing Emotions in Video Using Multimodal DNN Feature Fusion. In Proceedings of the Grand Challenge and Workshop on Human Multimodal Language (Challenge-HML), Melbourne, VIC, Australia, 18 July 2018; pp. 11–19. [CrossRef]
16. Majumder, N.; Hazarika, D.; Gelbukh, A.; Cambria, E.; Poria, S. Multimodal sentiment analysis using hierarchical fusion with context modeling. *Knowl. Based Syst.* **2018**, *161*, 124–133. [CrossRef]
17. Poria, S.; Cambria, E.; Hazarika, D.; Majumder, N.; Zadeh, A.; Morency, L.-P. Context-Dependent Sentiment Analysis in User-Generated Videos. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Vancouver, BC, Canada, 30 July–4 August 2017; pp. 873–883. [CrossRef]
18. Zadeh, A.; Liang, P.P.; Mazumder, N.; Poria, S.; Cambria, E.; Morency, L.-P. Memory Fusion Network for Multi-view Sequential Learning. *Proc. AAAI Conf. Artif. Intell.* **2018**, *32*, 1. [CrossRef]
19. Liang, P.P.; Liu, Z.; Zadeh, A.; Morency, L.-P. Multimodal Language Analysis with Recurrent Multistage Fusion. *arXiv* **2018**, arXiv:1808.03920.
20. Wang, Y.; Shen, Y.; Liu, Z.; Liang, P.P.; Zadeh, A.; Morency, L.-P. Words Can Shift: Dynamically Adjusting Word Representations Using Nonverbal Behaviors. *Proc. AAAI Conf. Artif. Intell.* **2019**, *33*, 7216–7223. [CrossRef]
21. Delbrouck, J.-B.; Tits, N.; Brousmiche, M.; Dupont, S. A Transformer-based joint-encoding for Emotion Recognition and Sentiment Analysis. *arXiv* **2020**, arXiv:2006.15955.
22. Emotions and Types of Emotional Responses, Verywell Mind. Available online: https://www.verywellmind.com/what-are-emotions-2795178 (accessed on 6 June 2023).
23. Torkamaan, H.; Ziegler, J. Exploring chatbot user interfaces for mood measurement: A study of validity and user experience. In Adjunct Proceedings of the 2020 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2020 ACM International Symposium on Wearable Computers, Virtual Event, 12–17 September 2020; pp. 135–138.
24. Torkamaan, H.; Ziegler, J. Mobile mood tracking: An investigation of concise and adaptive measurement instruments. Proceedings of the ACM on Interactive, Mobile. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2020**, *4*, 155. [CrossRef]

25. Dahmane, M.; Alam, J.; St-Charles, P.-L.; Lalonde, M.; Heffner, K.; Foucher, S. A Multimodal Non-Intrusive Stress Monitoring from the Pleasure-Arousal Emotional Dimensions. *IEEE Trans. Affect. Comput.* **2020**, *13*, 1044–1056. [CrossRef]
26. Universal Emotions, Paul Ekman Group. Available online: https://www.paulekman.com/universal-emotions/ (accessed on 6 June 2023).
27. Niwlikar, A.B. Popular Theory of the 6 Basic Emotions by Paul Ekman, Careershodh, 25 February 2022. Available online: https://www.careershodh.com/popular-theory-of-the-6-basic-emotions-by-paul-ekman/ (accessed on 6 June 2023).
28. Lim, N. Cultural differences in emotion: Differences in emotional arousal level between the East and the West. *Integr. Med. Res.* **2016**, *5*, 105–109. [CrossRef] [PubMed]
29. Emotion-Detecting Tech Should Be Restricted by Law—AI Now, BBC News. Available online: https://www.bbc.com/news/technology-50761116 (accessed on 5 June 2023).
30. Nast, C. Job Screening Service Halts Facial Analysis of Applicants, Wired, 12 January 2021. Available online: https://www.wired.com/story/job-screening-service-halts-facial-analysis-applicants/ (accessed on 6 June 2023).
31. Cogito—AI for a Better Human Customer Service Experience, Digital Innovation and Transformation. Available online: https://digital.hbs.edu/platform-digit/submission/cogito-ai-for-a-better-human-customer-service-experience/ (accessed on 6 June 2023).
32. Vincent, J. AI 'Emotion Recognition' Can't Be Trusted, The Verge, 25 July 2019. Available online: https://www.theverge.com/2019/7/25/8929793/emotion-recognition-analysis-ai-machine-learning-facial-expression-review (accessed on 6 June 2022).
33. Tatan, V. Understanding CNN (Convolutional Neural Network), Medium, 23 December 2019. Available online: https://towardsdatascience.com/understanding-cnn-convolutional-neural-network-69fd626ee7d4 (accessed on 6 June 2023).
34. Dobilas, S. LSTM Recurrent Neural Networks—How to Teach a Network to Remember the Past, Medium, 5 March 2022. Available online: https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e (accessed on 6 June 2023).
35. What is Supervised Learning? | IBM. Available online: https://www.ibm.com/cloud/learn/supervised-learning (accessed on 6 June 2023).
36. Text-Emotion-Detection. Available online: https://www.kaggle.com/dataset/f10c38f8f356a43b344ca82476b6b32b5d31b99af19276ba1f7846004c0851f2 (accessed on 5 June 2023).
37. RAVDESS Emotional Speech Audio. Available online: https://www.kaggle.com/uwrfkaggler/ravdess-emotional-speech-audio (accessed on 5 June 2023).
38. CK+48 5 Emotions. Available online: https://www.kaggle.com/gauravsharma99/ck48-5-emotions (accessed on 6 June 2023).
39. Kumar, S.; Bhuyan, M.K.; Chakraborty, B.K. Extraction of informative regions of a face for facial expression recognition. *IET Comput. Vis.* **2016**, *10*, 567–576. [CrossRef]
40. Facial Expression Dataset Image Folders (fer2013). Available online: https://www.kaggle.com/datasets/astraszab/facial-expression-dataset-image-folders-fer2013 (accessed on 25 October 2023).
41. Wang, X.; Wang, K.; Lian, S. A survey on face data augmentation for the training of deep neural networks. *Neural Comput. Appl.* **2020**, *32*, 15503–15531. [CrossRef]
42. Zhang, H.; Gou, R.; Shang, J.; Shen, F.; Wu, Y.; Dai, G. Pre-trained deep convolution neural network model with attention for speech emotion recognition. *Front. Physiol.* **2021**, *12*, 643202. [CrossRef]
43. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
44. CMU-MOSEI Dataset. Available online: http://multicomp.cs.cmu.edu/resources/cmu-mosei-dataset/ (accessed on 24 October 2023).
45. Revina, I.; Emmanuel, W. A survey on human face expression recognition techniques. *J. King Saud Univ. Comput. Inf. Sci.* **2021**, *33*, 619–628. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

PRC-Light YOLO: An Efficient Lightweight Model for Fabric Defect Detection

Baobao Liu [1,2,*], Heying Wang [1], Zifan Cao [1], Yu Wang [1], Lu Tao [1], Jingjing Yang [1] and Kaibing Zhang [1]

[1] School of Computer Science, Xi'an Polytechnic University, Xi'an 710048, China; whey820@163.com (H.W.); zifan_96@163.com (Z.C.); w201572972@163.com (Y.W.); yangj012019@163.com (J.Y.); zhangkaibing@xpu.edu.cn (K.Z.)

[2] Shaanxi Key Laboratory of Clothing Intelligence, School of Computer Science, Xi'an Polytechnic University, Xi'an 710048, China

* Correspondence: liubaobao1222@163.com

Abstract: Defect detection holds significant importance in improving the overall quality of fabric manufacturing. To improve the effectiveness and accuracy of fabric defect detection, we propose the PRC-Light YOLO model for fabric defect detection and establish a detection system. Firstly, we have improved YOLOv7 by integrating new convolution operators into the Extended-Efficient Layer Aggregation Network for optimized feature extraction, reducing computations while capturing spatial features effectively. Secondly, to enhance the performance of the feature fusion network, we use Receptive Field Block as the feature pyramid of YOLOv7 and introduce Content-Aware ReAssembly of FEatures as upsampling operators for PRC-Light YOLO. By generating real-time adaptive convolution kernels, this module extends the receptive field, thereby gathering vital information from contexts with richer content. To further optimize the efficiency of model training, we apply the HardSwish activation function. Additionally, the bounding box loss function adopts the Wise-IOU v3, which incorporates a dynamic non-monotonic focusing mechanism that mitigates adverse gradients from low-quality instances. Finally, in order to enhance the PRC-Light YOLO model's generalization ability, we apply data augmentation techniques to the fabric dataset. In comparison to the YOLOv7 model, multiple experiments indicate that our proposed fabric defect detection model exhibits a decrease of 18.03% in model parameters and 20.53% in computational load. At the same time, it has a notable 7.6% improvement in *mAP*.

Keywords: fabric defect detection; YOLOv7; lightweight network; HardSwish; Wise-IOU v3

1. Introduction

The detection of fabric defects plays a significant role in the industrial sector. In the process of fabric manufacturing, the defects on the surface of fabric are inevitable. Therefore, scholars are focused on the effective identification and precise localization of these fabric defects [1].

There are two main categories of methods for detecting fabric defects: traditional methods and deep learning methods. The traditional methods include the texture structure method, histogram statistics, spectral method, modeling method and adaptive dictionary learning method for fabric defect detection. Li et al. put forward an approach that relies on saliency region and similarity localization detection to achieve accurate contour segmentation of periodic textured fabric images and address the challenge of defect detection [2]. This detection speed is not dominant, and the detection effect for fabric images with no obvious defects or complex texture fabric images is generally not satisfactory. Xiang et al. proposed a yarn-dyed fabric defect detection algorithm based on Fourier convolution and convolutional self-encoder that avoids the conventional method of introducing noise in the training phase and employs a random masking method to generate image pairs for training instead [3]. In the process, some spatial domain information may be lost, making it difficult

for the decoder to fully recover the original input when inverting the frequency domain data back to the spatial domain. Wu et al. developed a method of cotton fabric defect detection based on K-Singular Value Decomposition dictionary learning [4], in which the dictionary is generally better suited to handle sparse textures. Kanwal et al. utilized the K-means algorithm to develop a bag-of-words model and proposed a significance-based bag-of-words model for fabric defect detection [5]. The bag-of-words model may ignore the spatial information and contextual relationships of features, leading to the loss of local information and structural information in images. Although the detection speed and performance of the methods based on traditional image processing are better than those of manual detection, the methods of generalization ability are inadequate. In the past two decades, deep learning has become a trend for fabric defect detection, and the techniques have achieved significant advancements in this field [6]. Methods for detection based on deep learning can be classified further into two primary categories: single-stage and two-stage methods. The single-stage methods mainly include Solid State Drive (SSD) [7] and You Only Look Once (YOLO) [8,9] series, and the two-stage algorithms comprise R-CNN [10], Fast R-CNN [11], Faster R-CNN [12] and so on. Zhao et al. utilized ROI Align and ResNet50 network to replace the region of interest pooling layer and VGG16 features extraction network in Faster R-CNN [13]. In addition, the softmax classifier is used to identify the image and obtain the prediction results, but the detection speed needs to be improved. Zhang et al. introduced the channel attention mechanism to highlight defect features and suppress background noise features in MobileNetV2-SSDLite and redefined the loss function by Focal Loss [14]. However, this model still has a lot of parameters and is difficult to deploy. Based on YOLOv5s, Zhou et al. used K-Means++ clustering to acquire the anchor frames, followed by combining the CARAFE upsampling operator. In addition, they added a small target detection layer to the baseline model, which results in an increase in the amount of model parameters [15]. Wu et al. embedded Classification-Aware Regression Loss (CARL) into YOLOF. Although this method establishes a relationship between classification and target localization [16], CARL exhibits sensitivity to noise and outliers within the data, potentially resulting in the model's tendency to overfit to these anomalies. Lin et al. proposed a weighted bidirectional feature network by substituting the Backbone module within the original YOLOv5 structure with the Swin Transformer [17]. Moreover, they employed the generalized focal loss to decrease the missed detection rate and improve the learning of positive sample instances. However, the Swin Transformer increases the parameter number and calculation amount of the original network, and there is no guarantee that the model will achieve the same detection results on either mobile or embedded terminals. Guo et al. embedded the Atrous Spatial Pyramid Pooling module and squeeze-and-excitation channel attention module into YOLO. They used convolution kernels with varying expansion rates to pool feature maps and achieved improved defect detection accuracy by self-learning the weights of each feature channel [18], but the background of the dataset is too simple, which can lead to idealized experimental results. Di et al. developed a deconvolution-based adaptive feature fusion network and introduced a context receptive field block into the Backbone of the YOLOv5 model; in addition, they used exponential distance IOU as the model bounding box loss function [19]. The detection precision of the model is improved by adaptively weighting the bounding box gradient, but this method has 3.8M more parameters than the baseline YOLOv5 and a lower *mAP* value than YOLOv7.

The methods mentioned above can achieve higher defect detection accuracy or the quantity of computation, but often at the cost of adding more parameters to the original model. In addition, the background of the dataset used in some literature is not complicated enough, so they have high *mAP* values. In this paper, we made some data images of fabric defects with complicated background and researched the structure of YOLOv7 [20]. Our main contributions are as follows:

- We propose Partial Convolution (PConv) to replace the 3×3 Conv of Extended-Efficient Layer Aggregation Networks (E-ELAN) in the Backbone module. This replacement aims to decrease network model calculation and parameters.
- In the Neck module, we incorporate a Receptive Field Block (RFB) [21] and integrate the Content-Aware ReAssembly of FEatures (CARAFE) lightweight upsampling operator.
- We utilize the HardSwish activation function to decrease the computational burden and memory access of the network. The Wise-IOU v3 with dynamic non-monotonic focusing mechanism is applied as the bounding box loss function of the model.
- Compared with YOLOv7 and other detection models, the experimental results validate that PRC-Light YOLO has the highest *mAP* and improves the performance of fabric defect detection.

The remaining sections of this article are organized as follows: In Section 2, we introduce the YOLOv7 model and its image processing process. Section 3 presents the PRC-Light YOLO model, which encompasses the construction of lightweight modules, feature fusion network and optimization of activation functions and loss functions. Section 4 provides a detailed account of the experimental results and analysis. In Section 5, we design a fabric defect detection system. Section 6 discusses further research based on the experimental results. Finally, a conclusion is provided.

2. YOLOv7 Model Structure

The YOLOv7 model is mainly composed of three parts: Backbone, Neck and Head. The model structure is shown in Figure 1.

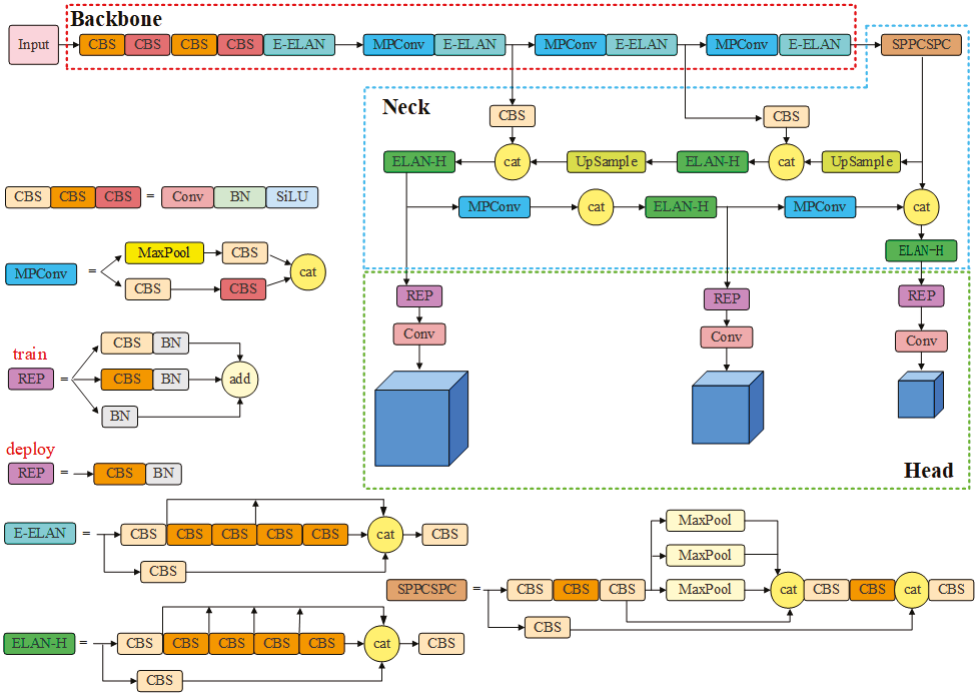


Figure 1. The network structure of YOLOv7.

2.1. Backbone

The Backbone part mainly comprises the CBS (Convolution, Batch normalization, SiLU) layer, E-ELAN module, and MPConv (Max Pool Convolution) module [22]. Among them, different CBS colors signify varied convolution kernels. The MPConv module comprises upper and lower branches. The upper branch shows the feature map via MaxPool operation, effectively reducing spatial dimensions by half. Subsequently, a CBS operation is applied to halve channel count. In the lower branch, the feature map is processed by the first CBS to reduce the number of channels, followed by another CBS to

further reduce channel count. Lastly, the upper and lower branches are merged by tensor splicing. The E-ELAN module contains two branches [23], and this module improves the ability of the network to learn additional features.

2.2. Neck

The Path Aggregation Feature Pyramid Network (PAFPN) structure is used in the Neck module of the YOLOv7 for feature fusion [24]. This involves transmitting and fusing higher-level feature information through an upsampling process, followed by feature map predictions obtained through a down-sampling fusion method. The final output comprises results from three feature layers. As image processing may result in significant image distortion, the SPPCSPC module incorporates three parallel MaxPool operations during a series of convolutional operations, effectively reducing the occurrence of image distortion.

2.3. Head

In the Head detection section, the REP module differs between the training and inference stages. During the training stage, it comprises three branches. While in the inference stage, the REP module undergoes reparameterization and consists of only a 3×3 Conv.

Thus, the fabric defect detection process based on YOLOv7 is outlined as follows. Firstly, preceding the input of an image into the Backbone, the model executes a series of operations, including adaptive image scaling, Mosaic data enhancement, adaptive anchor frame calculation and so on. Secondly, the Backbone module extracts features from the processed image. Thirdly, the Neck module conducts feature fusion processing on the extracted features from the Backbone, generating feature maps of three different sizes: large, medium and small. Finally, the Head layer detects the fused features and outputs the final results.

3. Fabric Defect Detection Based on PRC-Light YOLO

Although YOLOv7 has excellent detection accuracy and speed on datasets such as COCO and PASCAL VOC [20], its performance diminishes for fabric defect detection in fabric datasets. Thus, we study the theoretical foundation and network architecture of YOLOv7, subsequently proposing PRC-Light YOLO for a fabric defect detection model.

3.1. Lightweight Backbone Network

The reduction of floating-point operations (*FLOPs*) can alleviate the computational burden of neural networks, thereby shortening both forward and backward propagation times. This reduction significantly contributes to achieving lower latency. Simultaneously, lower *FLOPs* can enhance the parallelism of neural networks on hardware, increasing throughput and enabling faster processing of input data streams. Consequently, to achieve lightweight neural networks that meet the demands of low latency and high throughput, the focus of most researchers remains on minimizing the total number of *FLOPs*. This is because there exists the following relationship between neural network latency and *FLOPs*:

$$Latency = \frac{FLOPs}{FLOPS} \tag{1}$$

Floating-point operations per second (*FLOPS*) is a metric for computing speed efficiency. As indicated by Equation (1), reducing the number of *FLOPs* does not always guarantee a decrease in neural network latency. Hence, it is essential to concurrently reduce *FLOPs* and optimize *FLOPS* to truly achieve low latency.

Chen et al. demonstrated that frequent memory accesses by convolution operations reduced *FLOPS* values in neural networks [25]. ShuffleNet [26], MobileNet [27] and GhostNet [28], for example, embed Group Convolution (GConv) or Depthwise Convolution (DWConv) to extract spatial features. For an input $I \in R^{c \times h \times w}$, with c convolution kernels $W \in R^{k \times k}$, we obtain an output $O \in R^{c \times h \times w}$. Then, the *FLOPs* for regular convolution

is $h \times w \times k^2 \times c^2$, and the DWConv's *FLOPs* is $h \times w \times k^2 \times c$. Despite the lower *FLOPs*, the DWConv operation cannot be directly followed by the regular convolution operation, or it will lead to a significant degradation of model accuracy. To compensate for the drop in precision, the PointWise Convolution (PWConv) operation is typically employed after DWConv, with the channel count in DWConv being augmented to c. As a result, the Memory Access Cost (MAC) for DWConv is as follow:

$$h \times w \times 2c' + k^2 \times \mathrm{c}' \approx h \times w \times 2c' \tag{2}$$

The MAC for regular convolution can be represented as follows:

$$h \times w \times 2c' + k^2 \times \mathrm{c} \approx h \times w \times 2c \tag{3}$$

A comparison of Equations (2) and (3) suggests that the DWConv operation may elevate the frequency of memory accesses in the process of reducing *FLOPs*. Furthermore, since these models typically involve extra data processing operations such as splicing, disambiguation, and pooling, the inclusion of DWConv does not invariably curtail the neural network's latency.

Therefore, we adopt a novel partially convolution PConv in the Backbone, which reduces the *FLOPs* and MAC of neural networks. The principle of the PConv is illustrated in Figure 2.

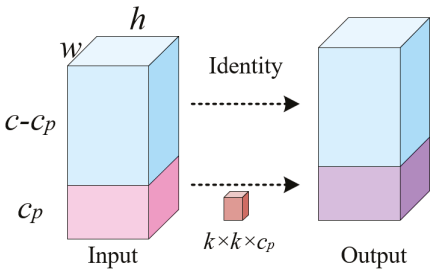


Figure 2. The principle of PConv.

As can be seen from Figure 2, PConv only performs the convolution operation on part of the input channels by using the regular conventional Conv for spatial feature extraction. Meanwhile, the remaining $c - c_p$ channels keep an identity mapping to Output. This approach allows selective processing while preserving certain channel information. The *FLOPs* of PConv is calculated to be $h \times w \times k^2 \times c_p^2$, and the MAC is $h \times w \times 2c_p + k^2 \times c_p^2 \approx h \times w \times 2c_p$, where c_p denotes the number of channels in PConv that perform regular convolution. The c_p is usually taken to be $1/4c$. Therefore, the *FLOPs* of PConv are only 1/16 of those of the regular Conv, resulting in reduced memory access. This paper replaces all 3×3 convolution operations within E-ELAN of the YOLOv7 Backbone network with PConv. This substitution allowed for a reduction in the frequency of memory accesses by the neural network. This substitution allowed for the maintenance of higher *FLOPS* with fewer *FLOPs*, consequently reducing neural network latency and making the network more effective at extracting spatial features.

3.2. Improved Feature Fusion Network

3.2.1. RFB Feature Pyramid

The Feature Pyramid SPPCSP is made up of two components: Single-Point Perfection (SPP) [29] and Complexity-Sensitive Perfection (CSP) [30]; as a result, it has a larger number of model parameters and computations, which leads to slower inference speed of the YOLOv7 model. Additionally, in the sampling process, SPPCSP might cause a reduction in the size of input feature maps. This could potentially lead to a degree of feature information loss, which has an impact on the detection of small objects. Consequently, we integrate the RFB feature pyramid into the Neck of the YOLOv7 model, which is able to learn deeper

features from the lightweight CNN model, making the detection model faster and more accurate with less calculation.

The RFB is a multi-branch convolution [31] block inspired by Inception [32] in network construction. Its internal structures are divided into two parts: multi-branch convolution employing various convolution kernels and dilated convolutions featuring variable rates. The dilated convolution operation's primary function is to expand the receptive field so that it can capture crucial information in regions with more context. The structure of the RFB is illustrated in Figure 3.

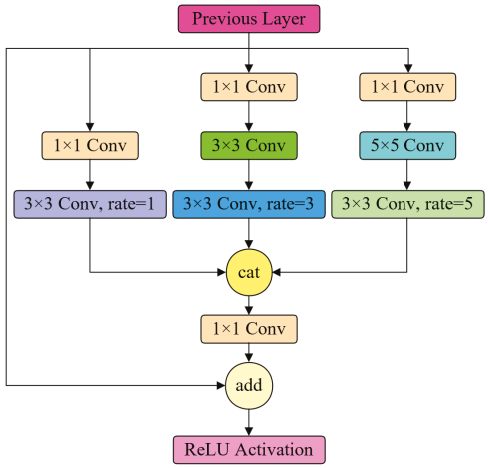


Figure 3. The structure of the RFB.

In the RFB, we utilize branches with varying dilation factors to capture multiscale information and different ranges of dependencies. Despite these branches having distinct receptive fields, they share the same weights, which help reduce the number of parameters, thereby lowering the risk of overfitting and maximizing the utilization of information from each sample. Finally, the RBF concatenates the computation results from different branches, preventing the model from encountering gradient explosion and vanishing gradient issues during training, ultimately achieving the goal of fusing different features.

3.2.2. CARAFE Upsampling Operator

The upsampling method in the YOLOv7 feature fusion network is the nearest neighbor interpolation method [33]. The principle of this method is to identify the nearest neighbor pixel in the target image based on each pixel's position in the original image and subsequently replace the value of that pixel in the original image with the value of the nearest neighbor pixel. However, the image quality after scaling by this method is poor, and obvious jagged edges may appear. To address this issue, we introduce the CARAFE upsampling operator. This innovation significantly bolsters the neural network's semantic feature extraction capacity during the upsampling process within object detection tasks. Importantly, this improvement is made without significantly increasing the amount of computations and parameters. The CARAFE upsampling process is depicted in Figure 4.

The CARAFE architecture is made of two primary modules: the upsampling kernel prediction module and feature reassembly module. These components work together to enable effective upsampling and feature reassembly within the network. In the upsampling kernel prediction module, assuming the upsampling ratio is σ, the process begins with channel compression through a 1×1 convolution operation on the input feature map of size $H \times W \times C$, resulting in a feature map of size $H \times W \times C_m$. Subsequently, encoding the feature map using a $k_{enc} \times k_{enc}$ convolutional kernel generates the recombined kernel $k_{up} \times k_{up}$, obtaining a feature map with $\sigma^2 \times k_{up}^2$ channels. The channels are then spatially unfolded and subjected to softmax normalization, producing a feature map of size $\sigma H \times \sigma W \times K_{up}^2$, where each feature value corresponds to an upsampling kernel. In the feature recombination module, through the mapping relationship, the region $k_{up} \times k_{up}$

centered in the input feature map is taken out, and then the dot product operation is performed with the upsampling kernel at the same position in $\sigma H \times \sigma W \times K_{up}^2$ to obtain the output value.

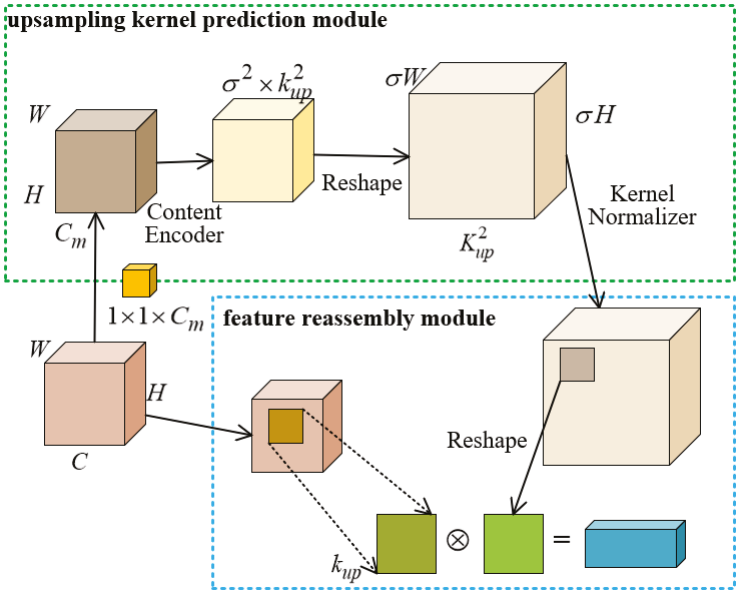


Figure 4. Upsampling process on CARAFE.

The CARAFE achieves instance-specific content-aware processing by generating real-time adaptive convolution kernels and can aggregate contextual information over a larger receptive field. Therefore, we introduce the CARAFE upsampling operator to replace two upsampling operations in the Neck structure of the YOLOv7 model. According to the trade-off relationship between k_{enc} and k_{up}, to maintain the performance and efficiency, k_{up} takes the value of 7 and k_{enc} takes the value of 5 in the experiments.

3.3. HardSwish Activation Function

Activation functions are significant in neural networks, introducing non-linearity and enabling the networks to learn and represent complex functions, thereby enhancing the network's representational capacity [34]. In the absence of activation functions, deeper neural networks would be limited to handling linearly differentiable problems [35]. Commonly used activation functions are Sigmoid, ReLU, Swish, Mish, GELU and so on. The YOLOv7 model employs the SiLU activation function, which is expressed as follows:

$$SiLU(x) = x \cdot sigmoid(x) = \frac{x}{1 + e^{-x}} \tag{4}$$

The computation process of the Sigmoid function involves exponentiation, which undoubtedly increases the computational complexity of the SiLU activation function. However, the Sigmoid activation function can be approximated using a piecewise linear function called HardSigmoid [36], which significantly reduces the computational cost. The HardSigmoid activation function and HardSwish activation function expressions are as follows:

$$HardSigmoid(x) = \begin{cases} 0, & x \leq -3 \\ 1, & x \geq 3 \\ (x+3)/6, & otherwise \end{cases} \tag{5}$$

$$HardSwish(x) = \begin{cases} 0, & x \leq -3 \\ (x^2+3x)/6, & -3 < x < 3 \\ x, & x \geq 3 \end{cases} \tag{6}$$

The HardSwish and SiLU activation function output curves are shown in Figure 5. HardSwish and SiLU are differentiable at all points and have no upper bounds, enabling them to avoid overfitting while making the model more generalizable. Furthermore, the segmented function reduces the number of memory accesses, which significantly reduces the latency cost.

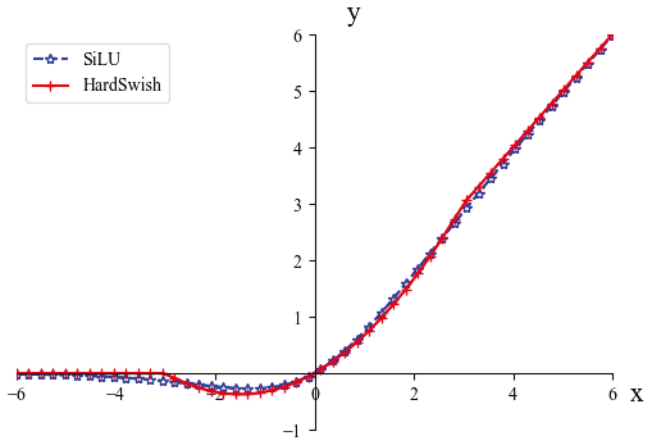


Figure 5. Activation function output curves.

3.4. Wise-IOU v3 Bounding Box Loss

The loss function of YOLOv7 comprises three parts: Bounding Box Loss (BoxLoss), Classification Loss (ClsLoss) and Object Confidence Loss (ObjLoss). Among them, the BoxLoss employs the Complete Intersection Over Union (CIOU) loss [37], and the ObjLoss and ClsLoss are calculated using the binary cross-entropy loss function. The YOLOv7 total loss expression is shown in Equation (7).

$$Loss = Loss_{Obj} + Loss_{cls} + Loss_{Box} \tag{7}$$

The CIOU loss in YOLOv7 considers three geometric factors: the overlap area between bounding boxes, the difference in aspect ratios and the distance between their center points. Given a target frame and a prediction frame, the CIOU loss is calculated as:

$$L_{CIOU} = 1 - IOU + \frac{\sqrt{(i_x - j_x)^2 + (i_y - j_y)^2}}{c^2} + \alpha v \tag{8}$$

$$\alpha = \frac{v}{(1 - IOU) + v} \tag{9}$$

$$v = \frac{4}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h})^2 \tag{10}$$

The gradient of v with respect to w and h is calculated as follows:

$$\begin{cases} \frac{\partial v}{\partial w} = \frac{8}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}) * \frac{h}{w^2+h^2} \\ \frac{\partial v}{\partial h} = \frac{8}{\pi^2}(\arctan\frac{w^{gt}}{h^{gt}} - \arctan\frac{w}{h}) * \frac{w}{w^2+h^2} \end{cases} \tag{11}$$

(i_x, i_y), (j_x, j_y) are the center point coordinates of the real frame and the prediction frame, respectively, v measures the similarities in length ratios, α represents the balancing parameter and the diagonal length of the smallest outer rectangle containing the true and predicted boxes is denoted by c. Analysis of Equation (10) shows that when the aspect ratios of the predicted and real frame are equal, the penalty term related to the aspect ratio becomes 0. Thus, the aspect ratio penalty term loses its effect, which reduces the optimization performance of the model for similarity. From Equation (11), the relationship between $\frac{\partial v}{\partial w}$ and $\frac{\partial v}{\partial h}$ is inversely proportional, and during training, the simultaneous adjustment of both width and height dimensions in the prediction frame is not feasible. This constraint

hinders the prediction frame from effectively approximating the dimensions of the real frame [38].

The CIOU employs a monotonic focus mechanism, which is predicated on the assumption that high-quality examples exist within the training data. Its primary objective is to enhance the fitting capability of bounding box loss. Nevertheless, when the target detection training dataset includes low-quality examples, indiscriminately emphasizing the regression of bounding boxes for these low-quality instances evidently leads to a deterioration in the model's detection performance [39]. Therefore, we employ Wise-IOU v3 as the bounding box loss, which incorporates a non-monotonic dynamic focusing mechanism. This mechanism relies on the outlier degree rather than IOU to assess the quality of anchor boxes, which diminishes the detrimental gradient caused by low-quality examples, mitigates the competitiveness of high-quality anchor boxes and improves the overall performance of the detector. The equations are as follows:

$$R_{WIOU} = \exp\left(\frac{(i_x - i_y)^2 + (j_x - j_y)^2}{(W^2 + H^2)^*}\right) \tag{12}$$

$$L_{w1} = R_{WIOU} * L_{IOU} \tag{13}$$

$$r = \frac{\beta}{3 \times 1.9^{\beta-3}} \tag{14}$$

$$\beta = \frac{L^*_{IOU}}{\overline{L_{IOU}}} \in [0, +\infty) \tag{15}$$

$$L_{WIOU} = r * L_{w1} \tag{16}$$

W and H represent the width and height, respectively, of the smallest outer rectangle of the real frame and the prediction frame and $(W^2 + H^2)^*$ denotes the separation of W and H from the Computational Graph to prevent R_{WIOU} from making gradients that impede convergence. $L^{\gamma^*}_{IOU}$ is the monotonic focusing factor of L_{w1}. $\overline{L_{IOU}}$ denotes the sliding mean with momentum m, and its dynamic updates contribute to maintaining an elevated overall β, effectively resolving the issue of slow convergence in the later stages of training. This enables Wise-IOU to flexibly adjust gradient gain allocation strategies during the training phase in order to best adapt to the current situation.

3.5. PRC-Light YOLO Model Structure

In summary, this paper proposes PRC-Light YOLO. The differences between PRC-Light YOLO and YOLOv7 are as follows:

(1) We replace all 3×3 convolution operations of E-ELAN in the Backbone of YOLOv7 with PConv introduced in Section 3.1 to form the PE-ELAN module.

(2) The RFB feature pyramid proposed in Section 3.2.1 replaces the SPPCSP module in the Neck of YOLOv7.

(3) The CARAFE upsampling operators, as proposed in Section 3.2.2, replace the two upsampling operations in the Neck structure of the YOLOv7.

(4) Due to the YOLOv7 model's utilization of the SiLU activation function, which is composed of exponentials, there is an increased computational cost. To address this issue, we bring in the HardSwish activation function, which is composed of piecewise functions, significantly reducing the computational burden of the model. Consequently, we replace the SiLU activation function in the CBS module with the HardSwish activation function. Convolution, Batch Normalization and HardSwish activation functions constitute the CBH module.

(5) We have replaced the bounding box loss function of YOLOv7 with Wise-IOU v3. It helps mitigate adverse gradients stemming from low-quality instances, thereby enhancing the overall performance of the detector.

The PRC-Light YOLO structure is shown in Figure 6.

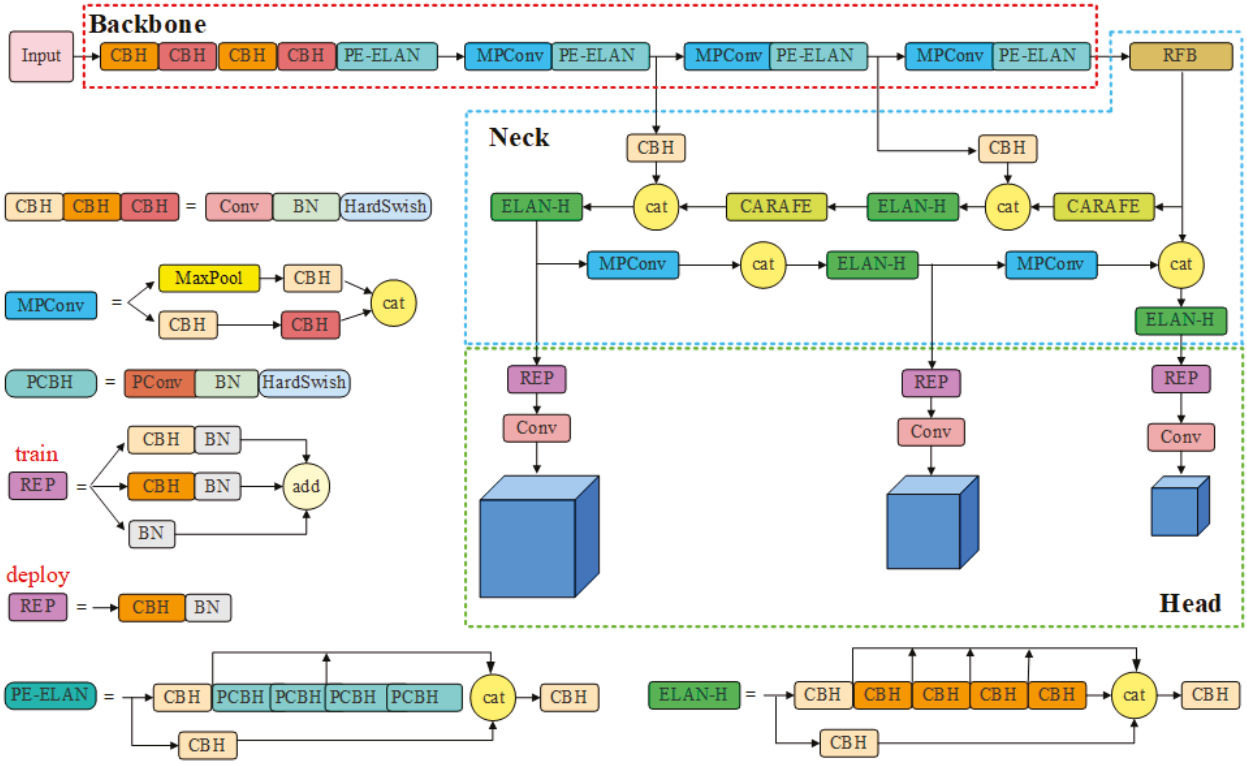


Figure 6. The PRC-Light YOLO structure.

4. Experiment

4.1. Fabric Defect Image Dataset

The dataset [40] was acquired from multiple resources, including apparel factories, the existing literature and the "Xuelang Manufacturing AI Challenge", from which we select a total of 1061 images encompassing 4 categories: Warp hanged, Yard defects, Holes and Stains, named as dj, ds, pd and wz, respectively. Their characteristics are as follows:

(1) Warp hanged is small in size, typically suspended above the fabric without proper weaving, making it difficult to discern against a darker background.

(2) Yard defect refers to the occurrence of defects along the horizontal direction of the fabric within a specific length.

(3) Hole refers to a large number of broken weft holes on the cloth surface and the emergence of weft yarn, which is a serious defect.

(4) Stain manifests as an area on the fabric with a distinct color different from its surroundings, varying in size from large to small within the dataset.

In order to reduce the unnecessary computations of the network model, all images are cropped to 200 pixel × 200 pixel to speed up the training. Then, we use MakeSense to label the defect types in the images and extract the txt annotation information files required for YOLOv7 model training. In order to enrich the fabric dataset, we randomly rotate the images, add noise, change the brightness, chromaticity and saturation techniques of the samples, perform data enhancement techniques on 1061 datasets, expand the dataset to 4244 datasets, enhance the PRC-Light YOLO model's generalization capacity and then divide the training set, validation set and testing set according to the ratio of 8:1:1. Parts of the samples are shown in Figure 7.

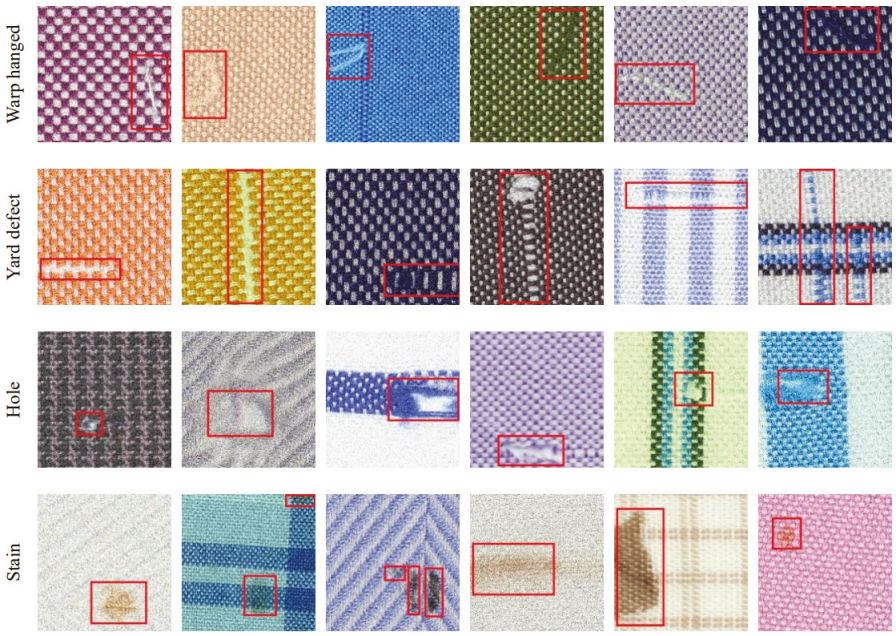


Figure 7. Defective fabric image samples.

4.2. Experimental Environment and Parameter Configuration

The experimental equipment parameters are shown in Table 1. The input image size is 320 × 320, with a total of 150 epochs of training with batch size 16; the initial learning rate is set to 0.01.

Table 1. Experimental equipment parameters.

Name	Operating System	RAM	Graphics Card	CUDA	Python	Framework
Parameter	Windows X64	12G	NVIDIA Quadro P4000	11.3	3.8	PyTorch

4.3. Evaluation Metrics

To verify the improved model's performance, we utilize Precision (*P*), Recall (*R*), *F1-score* and mean Average Precision (*mAP*) as the measures of detection accuracy; the formulas are calculated as follows:

$$P = \frac{TP}{TP + FP} \tag{17}$$

$$R = \frac{TP}{TP + FN} \tag{18}$$

$$F_1 - score = \frac{2 * P * R}{P + R} = \frac{2TP}{2TP + FP + FN} \tag{19}$$

$$mAP = \frac{\sum_{i=1}^{N} AP_i}{N} \tag{20}$$

TP represents true positives, representing the amount of positive samples accurately predicted to be positive in the positive samples. *FP* denotes false positives, indicating the count of negative samples that are predicted as positive samples. *FN* represents false negatives, depicting the count of positive samples that are wrongly predicted to be negative. Each category can draw PR curves based on Precision and Recall, every category corresponds to one PR curve, the area covered by which indicates the Average Precision (*AP*) of the category. *mAP* is the average value after summing up the *APs* of all categories.

4.4. Ablation Experiment

An ablation experiment is an important method in deep learning; by improving the modules of the YOLOv7 network structure and assessing the contribution of different improvement methods on its performance, we conducted seven sets of ablation experiments in all, where "✓" corresponds to the improvement method and the corresponding experimental order. The experimental results are listed in Table 2. Experiment 1 (exp1) is based on YOLOv7. Experiment 2 (exp2) replaces YOLOv7's SiLU activation function with HardSwish. Experiment 3 (exp3) incorporates the Wise-IOU v3 into YOLOv7 as the bounding box loss function. Experiment 4 (exp4) substitutes YOLOv7's SSPCSP pyramid with the RFB module. Experiment 5 (exp5) replaces the nearest neighbor interpolation method in YOLOv7 with the CARAFE operator. Experiment 6 (exp6) integrates HardSwish, Wise-IOU v3, RFB and CARAFE into YOLOv7. Experiment 7 (exp7) is based on PRC-Light YOLO, and it builds upon exp6 by replacing the 3×3 convolution operation in the Backbone's E-ELAN with PConv.

Table 2. Results of ablation experiments.

Exp	HardSwish	Wise-IOU v3	RFB	CARAFE	PConv	Params/M	*GFLOPs*/G	*mAP*/%	Inference Time/ms
exp1	×	×	×	×	×	37.21	105.2	75.4	35.08
exp2	✓	×	×	×	×	37.21	105.2	78.1	31.49
exp3	×	✓	×	×	×	37.21	105.2	79.1	32.08
exp4	×	×	✓	×	×	33.94	102.5	80.6	33.31
exp5	×	×	×	✓	×	37.87	106.5	79.8	34.65
exp6	✓	✓	✓	✓	×	34.6	103.8	81.1	30.62
exp7	✓	✓	✓	✓	✓	**30.5**	**83.6**	**83**	**28.46**

From Table 2, it can be seen that the first set of experiments is based on YOLOv7, the model parameter number is 37.21 M, the Giga FLoating-Point Operations Per Second (*GFLOPs*) is 105.2 G and the *mAP* is 75.4%, followed by subsequent experiments, which are all based on the results of the first set of experiments as a benchmark for comparison. The *GFLOPs* mentioned above is used to measure the computational complexity of a model, and $1 GFLOPs = 1 \times 10^9$ *FLOPs*, so lower a *GFLOPs* reduces the latency of the neural network, thereby improving their computational speed. Experiments 2 and 3, without increasing the parameters, respectively, increased the *mAP* values by 2.7% and 3.7%, and in experiment 2, by introducing the HardSwish activation function, the inference time is lower than the other ones that do not use the HardSwish activation function, which verifies that the characteristic of the segmentation function greatly reduces the computational cost of the model. In Experiment 4, the introduction of the RFB feature pyramid reduces the parameters by 8.79% compared to the SPPCSPC module in YOLOv7 while simultaneously improving the *mAP* by 5.2 percentage points. Experiment 5 utilizes the CARAFE lightweight upsampling module in the feature fusion module, and the *mAP* value goes to 79.8% with only a slight increase in parameters and *GFLOPs*, which is a significant improvement. In order to verify the lightweight effect of the PConv, Experiment 6 integrates HardSwish, Wise-IOU v3, RFB and CARAFE into YOLOv7. By comparing Experiment 7 with Experiment 6, Our findings reveal a reduction in the network's parameters by 11.85% and computations by 19.46%. Meanwhile, there is a 1.9% increase in the *mAP* value. In addition, the inference time of PRC-Light YOLO is 6.62 ms less than that of YOLOv7.

The variation curves of *mAP* values in seven groups of experiments are shown in Figure 8. Compared with YOLOv7, the number of parameters and *GFLOPs* of PRC-Light YOLO are reduced by 18.03% and 20.53% respectively, while the *mAP* value is improved by 7.6%. Experiments show that by improving the network structure of YOLOv7, the lightweight effect is significant; moreover, the efficacy of defect detection in fabrics using the enhanced model has significantly improved.

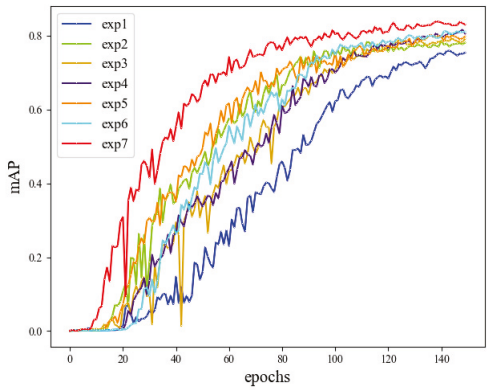
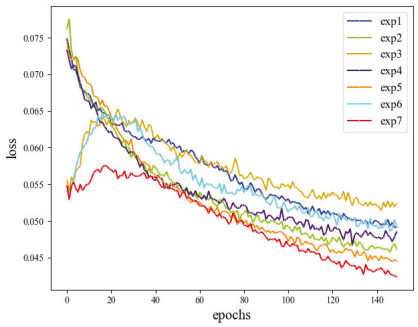
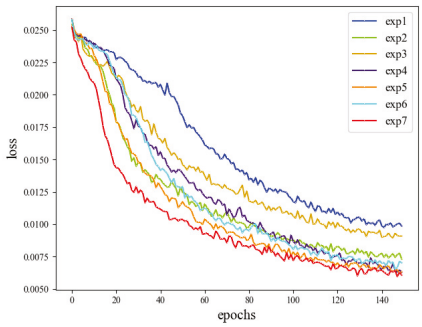


Figure 8. Change curve of *mAP* value.

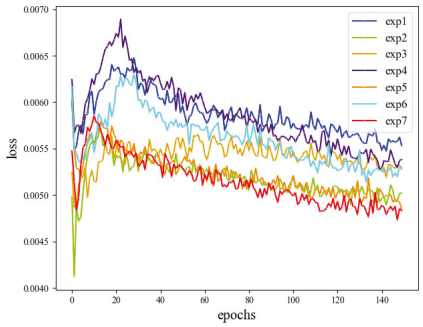
According to the loss curve chart above, it is evident that the PRC-Light YOLO model exhibits smaller loss values after convergence compared to other models. Specifically, experiments 3, 6 and 7 employ the Wise-IOU bounding box loss function, which dynamically adjusts the emphasis on different input regions over time. For anchor boxes with lower outliers, a smaller gradient boost is allocated, thereby directing the boundary regression towards normal-quality anchor boxes. This adjustment enhances the convergence speed of the network model. Figure 9 demonstrates that the PRC-Light YOLO model achieves lower loss values, thereby delivering superior performance in fabric defect detection.



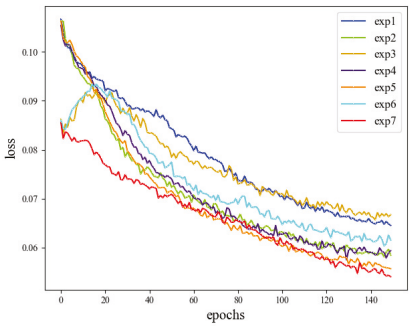
(**a**) Bounding box loss variation curve.



(**b**) Classification loss variation curve.



(**c**) Object confidence loss variation curve.

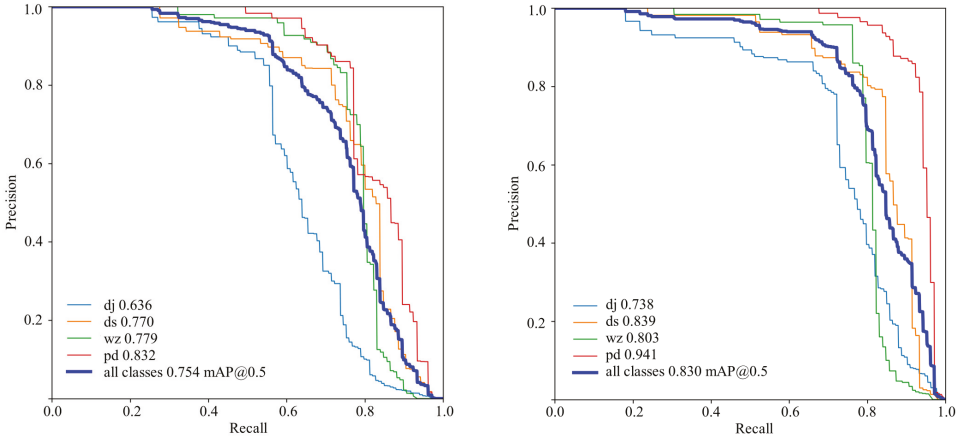


(**d**) The total loss function variation curve.

Figure 9. Loss Function. The loss function of the model consists of three parts, (**a**), (**b**) and (**c**), respectively, show the change curves of the BoxLoss, ClsLoss and ObjLoss for YOLOv7 and PRC-Light YOLO. (**d**) shows the total loss function variation curve.

According to Figure 10, it can be obtained that the *mAP* value of PRC-Light YOLO is increased from 75.4% at the beginning to 83%, and the experimental results verified

that the PRC-Light YOLO has improved the effect of defect detection. The *AP* values of threading, yard defects, stains and holes increased by 10.2%, 6.9%, 2.4% and 10.9%, respectively, with the most noticeable improvement in the detection of hanging warp and holes, and the *AP* values of the four categories of defects all are improved.



(**a**) PR curve for YOLOv7. (**b**) PR curve for PRC-Light YOLO.

Figure 10. Precision–Recall. (**a**) and (**b**), respectively, show the PR curves of YOLOv7 and PRC-Light YOLO on the fabric dataset.

4.5. Comparison of Detection Effect

Figures 11 and 12 show the YOLOv7 and PRC-Light YOLO effects of four different types of defect detection, including the location of the defect by box selection, as well as above the box to display the type of defect and confidence. From the comparison in the figure, it can be observed that the PRC-Light YOLO detection model has improved the accuracy of fabric defect detection.

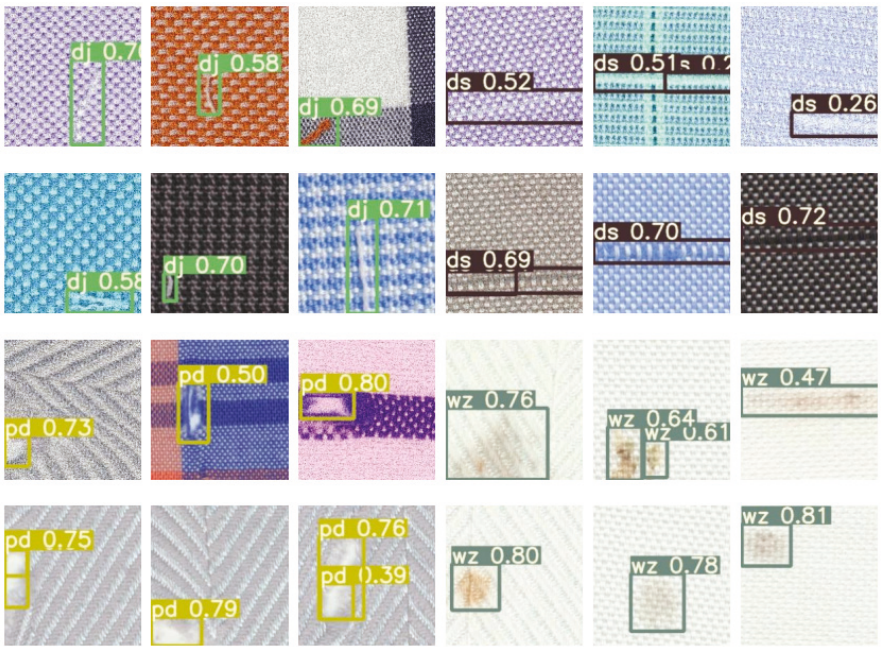


Figure 11. Detection effect of the YOLOv7.

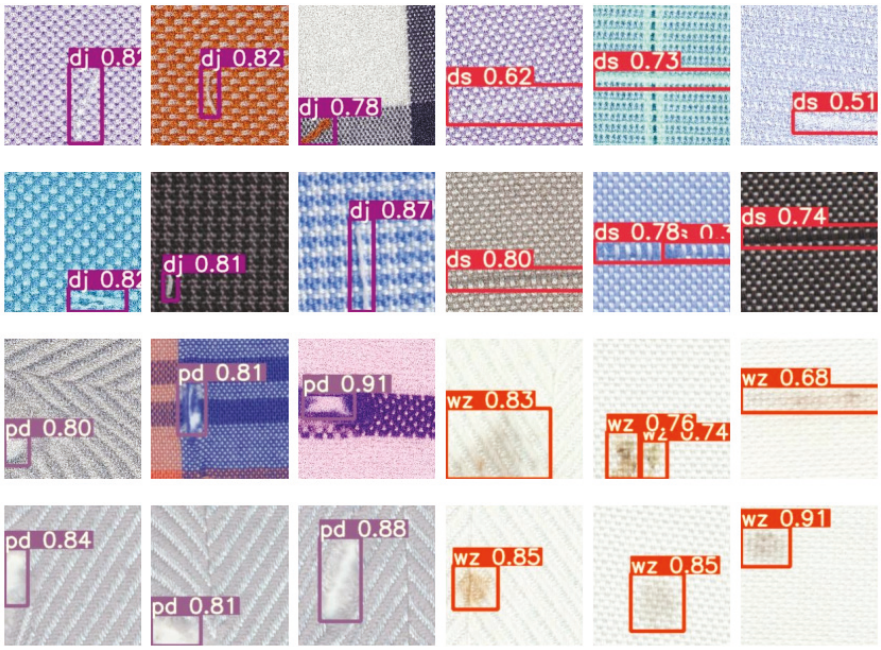


Figure 12. Detection effect of the PRC-Light YOLO.

4.6. Comparison Experiment

Through comparison experiments, the efficacy of various models can be assessed so that the better method can be selected. Therefore, we choose to conduct comparison experiments with Faster R-CNN, SSD, EfficientDet, CenterNet and YOLOv8x target detection models.

Table 3 shows a performance comparison of defect detection among various object detection models, with *P*, *R*, *F1-score*, *mAP* and inference time selected as evaluation metrics.

Table 3. Defect detectability of different models.

Models	*P*	*R*	*F1*	Params/M	*GFLOPs*/G	*mAP*/%	Inference Time/ms
Faster R-CNN	0.784	0.491	0.6	136.98	370.3	0.696	177.35
SSD	0.832	0.529	0.647	105.2	87.41	0.731	143.58
EfficientDet	0.787	0.492	0.605	52.11	34.97	0.697	22.71
CenterNet	0.838	0.547	0.662	125	69.66	0.756	158.2
YOLOv7	0.826	0.697	0.76	37.21	105.2	0.754	32.08
YOLOv8x	0.754	0.754	0.754	68.23	258.5	0.784	40.39
PRC-Light YOLO	**0.859**	**0.784**	**0.82**	**30.50**	**83.6**	**0.83**	**28.46**

Table 4 shows the comparative results of different target detection models for the fabric defect dataset.

Table 4. Detection effects of different models.

Categories	Faster R-CNN	SSD	EfficientDet	CenterNet	YOLOv7	YOLOv8x	PRC-Light YOLO
Warp hanged	0.738	0.756	0.731	**0.802**	0.636	0.732	0.738
Yard defects	0.582	0.596	0.599	0.652	0.77	0.768	**0.839**
Stain	0.692	0.759	0.686	0.765	0.779	0.786	**0.803**
Hotel	0.773	0.814	0.773	0.804	0.832	0.850	**0.941**

It can be seen that the two-stage detection model Faster R-CNN has the lowest *mAP* value compared to the other detection algorithms; in addition, the two-stage detection model has one more step of operation compared to the single-stage model: the candidate

regions are first generated, and then the regions are classified and identified, which also causes the slower training speed and inference time of Faster R-CNN. Compared with the target detection models of Faster R-CNN, SSD, EfficientDet, EfficientDet, YOLOv7 and YOLOv8x, the precision of PRC-Light YOLO is improved by 7.5%, 2.7%, 7.2%, 2.1%, 3.3% and 10.5%. The recall is increased by 29.3%, 25.5%, 29.2%, 23.7%, 8.7%, 3%; the *F1-score* is increased by 20%, 17.3%, 21.5%, 15.8%, 6%, 6.6%, and params is decreased by 77.7%, 71%, 41.5%, 75.6%, 18%, 55.3%. In terms of *GFLOPs*, PRC-Light YOLO is lower than Faster R-CNN, SSD, YOLOv7 and YOLOv8x but higher than EfficientDet and CenterNet, whereas in terms of *mAP*, PRC-Light YOLO outperforms EfficientDet and CenterNet by 13.3% and 7.4%. In terms of inference time, EfficientDet is faster than PRC-Light YOLO, but it is lower than PRC-Light YOLO in terms of P, R, F1 as well as *mAP* values, and PRC-Light YOLO is faster than all other models except EfficientDet. Comparing with other models, it can be found that PRC-Light YOLO is effective at detecting the Hole, and the *AP* value reaches 94.1%. In the case of Warp hanged, the detection effect of the PRC-Light YOLO model is not satisfactory, and the *AP* value is only higher than that of EfficientDet, YOLOv7 and YOLOv8x. Therefore, in the subsequent experiments, the PRC-Light YOLO model should be optimized by taking into full consideration the characteristics of the Warp hanged. By comparing the *AP* values of each model for detecting yard defects and stains, PRC-Light YOLO has at least 25.7% and 11.7% higher detection accuracy than the other models. Comprehensive indications reveal that our proposed PRC-Light YOLO has an outstanding detection performance on the fabric dataset, which validates the effectiveness of the improved method.

5. Fabric Defect Detection System

In order to further illustrate that the model has a certain value for engineering applications and provide a more intuitive depiction of the detection process and detection results, we use PyQt5 to design a fabric defect detection system based on PRC-Light YOLO. The UI interface of the system is shown in Figure 13.

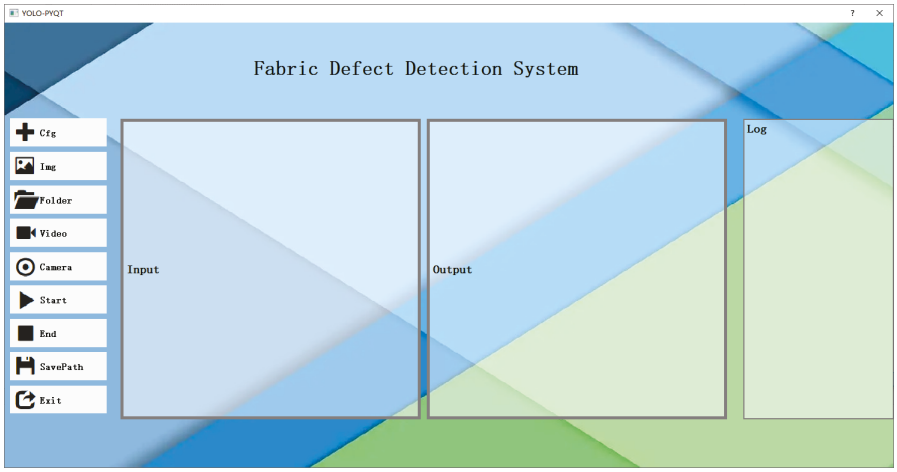


Figure 13. Detection system UI.

Clicking the "Cfg" button, select the necessary configuration file for the detection system. The file format should adhere to YAML format. The path to the chosen file will be displayed in the Statistical window. If the loading is successful, it will show "Load yaml success". The application effect of the YOLO model is shown in Figure 14.

The YAML file contents include the path to the weight file, the category of the blemish and the confidence threshold and IOU threshold. The confidence threshold serves to filter the model's output, displaying target boxes only when the detection probability exceeds this threshold. The IOU threshold primarily controls how the target detection algorithm handles overlapping targets. When two predicted bounding boxes intersect and both their

IOU values surpass the IOU threshold, the prediction box with lower confidence will be filtered out.

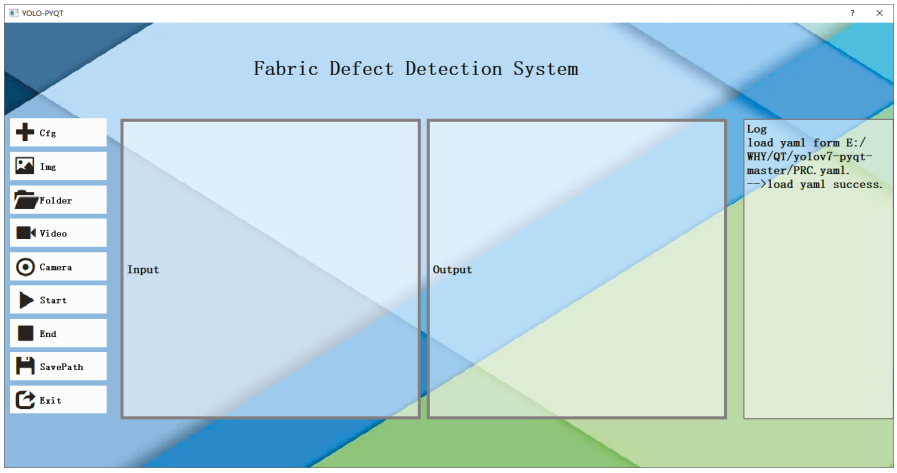


Figure 14. Application effect of the PRC-Light YOLO model.

After selecting the configuration file, you have the option to designate the path for saving the detection results. Additionally, the system is capable of processing images, videos and real-time data captured through a camera feed. The data path will be displayed in the Statistical window accordingly. By clicking the "Start Detection" button, the detection results will be presented to the right of the input image. The YOLO model detection effect is shown in Figure 15.

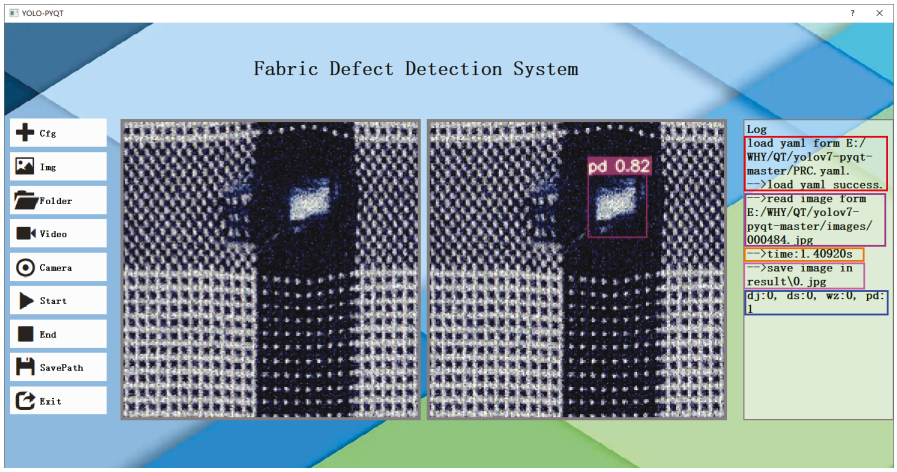


Figure 15. PRC-Light YOLO model checking effect.

In the Statistical window, you will also find information such as the time taken for detection, the path where the results are saved as well as the categories and quantities of defects detected. This information is essential for subsequent and timely manual inspection.

6. Discussion

The detection of defects in fabrics holds significant importance for ensuring fabric quality, reducing cost, enhancing production efficiency and preserving brand reputation. It is pivotal for the competitiveness and sustainable development of the fabric industry. This article presents a fabric defect detection model based on PRC-Light YOLO, which reduces the parameters of YOLOv7 while improving fabric defect detection accuracy. However, certain limitations still exist in some aspects.

(1) In the study of fabric defect detection, the size and variety of the dataset significantly impact the effectiveness of the model training. In practical industrial scenarios,

fabric backgrounds are often characterized by complexity and diversity and may also be affected by environmental factors such as light. However, the dataset used in the paper suffers from limitations such as a uniform background, limited quantity and insufficient diversity in defect types. Therefore, the exploration of methodologies for obtaining and constructing a substantial dataset of fabric defects becomes imperative. Furthermore, investigating fabric defects detection the background of intricate patterns and textures holds significant research significance so that the PRC-Light YOLO model can maintain robustness in real-world scenarios.

(2) In the research findings of this paper, it is evident that PRC-Light YOLO requires improvement in its capability to detect Warp hanged. Therefore, for the detection of minute defects, further investigation can be conducted into image preprocessing, network feature extraction capabilities or loss functions, with the aim of enhancing the precision of the network's detection capabilities.

(3) We have initially developed a textile defect detection system, which has interactivity and intuitiveness as its main goals. While this system introduces new perspectives for real-time fabric defect detection, it has not yet been integrated with hardware infrastructure. To achieve industrial applications, future efforts can involve further refining the detection platform and completing the setup of the system's hardware components. Additionally, we will collect user cases to evaluate the interactivity of the fabric defect detection system and continuously optimize it.

7. Conclusions

In response to the complex nature of defect types in fabrics, which vary in size, the computational intensity of detection models and the lower detection accuracy, this paper proposes the defect detection model PRC-Light YOLO. PRC-Light YOLO achieves network lightweight by replacing certain Conv layers with PConv layers in the Backbone. During the feature fusion stage, the RFB pyramid and the CARAFE upsampling operator are employed, and this approach enlarges the neural network's receptive field and enhances detection accuracy without significantly increasing the model's parameter count or computational load. PRC-Light YOLO adopts the HardSwish activation function, which exhibits piecewise characteristics that reduce computational costs and minimize memory access. Additionally, it utilizes Wise-IOU v3 as the bounding box loss function during model training, demonstrating robust performance in scenarios with varying dataset scales and highly imbalanced positive-to-negative sample ratios. Finally, we design a fabric defect detection system.

In order to improve the model's robustness and generalization ability, we extended the fabric dataset by applying diverse augmentation techniques, including random rotation, noise addition and adjustments to brightness, chromaticity and saturation. Ablation experimental results reveal that PRC-Light YOLO has a lower reduction in network model parameters by 18.03% and computations by 20.53% when compared with YOLOv7. Additionally, it showcases a 7.6% increase in *mAP*, enhancing the performance of detecting defects in fabrics. Furthermore, comparative experiments with other object detection models show that PRC-Light YOLO surpasses them in precision, recall, *F1-score* and *mAP* values. Although CenterNet outperforms PRC-Light YOLO by 6.4 percentage points in detecting Warp hanged, PRC-Light YOLO significantly outshines other models in detecting Yard defects, Stains and Holes with *AP* values consistently exceeding 80%. Notably impressive is the *AP* value of 94.1% achieved for detecting Holes.

To sum up, we propose the PRC-Light YOLO model, analyze the results of seven ablation experiments and seven comparative experiments, verify our model's excellent defect detection capability in fabrics and design a fabric defect detection system based on PRC-Light YOLO.

Author Contributions: Conceptualization, B.L. and H.W.; methodology, B.L. and H.W.; software, Z.C.; validation, H.W., Z.C. and Y.W.; investigation, H.W., L.T. and J.Y.; writing—original draft preparation, H.W.; writing—review and editing, H.W. and B.L.; supervision, B.L. and K.Z.; project administration and funding acquisition, B.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 61971339.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

YOLO	You Only Look Once
IOU	Intersection Over Union
mAP	mean Average Precision
SSD	Solid State Drive
R-CNN	Region with CNN
ROI	Region of Interest
VGG	Visual Geometry Group
CARAFE	Content-Aware ReAssembly of FEatures
PConv	Partial Convolution
E-ELAN	Extended-Efficient Layer Aggregation Networks
RFB	Receptive Field Block
CBS	Convolution, Batch normalization, SiLU
MPConv	Max Pool Convolution
PAFPN	Path Aggregation Feature Pyramid Network
REP	Re-Parameterization
FLOPs	Floating-Point Operations
FLOPS	Floating-Point Operations Per Second
DWConv	Depthwise Convolution
GConv	Group Convolution
PWConv	PointWise Convolution
MAC	Memory Access Cost
ReLU	Rectified Linear
GELU	Gaussian Error Linear Unit
SiLU	Sigmoid Weighted Liner Unit
ObjLoss	Object Confidence Loss
ClsLoss	Classification Loss
BoxLoss	Bounding Box Loss
CIOU	Complete Intersection Over Union
PE-ELAN	Partial convolution, Extended-Efficient Layer Aggregation Networks
CBH	Convolution, Batch normalization, HardSwish
P	Precision
R	Recall
F1	F1-score
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
AP	Average Precision
GFLOPs	Giga FLoating-Point Operations Per Second

References

1. Zhang, H.; Qiao, G.; Lu, S.; Yao, L.; Chen, X. Attention-based Feature Fusion Generative Adversarial Network for yarn-dyed fabric defect detection. *Text. Res. J.* **2023**, *93*, 1178–1195. [CrossRef]
2. Li, W.; Zhang, Z.; Wang, M.; Chen, H. Fabric Defect Detection Algorithm Based on Image Saliency Region and Similarity Location. *Electronics* **2023**, *12*, 1392. [CrossRef]
3. Xiang, J.; Pan, R.; Gao, W. Yarn-dyed fabric defect detection based on an improved autoencoder with Fourier convolution. *Text. Res. J.* **2023**, *93*, 1153–1165. [CrossRef]
4. Wu, J.; Li, P.; Zhang, H.; Su, Z. CARL-YOLOF: A well-efficient model for digital printing fabric defect detection. *J. Eng. Fibers Fabr.* **2022**, *17*, 15589250221135087. [CrossRef]
5. Kanwal, M.; Riaz, M.M.; Ali, S.S.; Ghafoor, A. Saliency-based fabric defect detection via bag-of-words model. *Signal Image Video Process.* **2023**, *17*, 1687–1693. [CrossRef]
6. Kahraman, Y.; Durmuşoğlu, A. Deep learning-based fabric defect detection: A review. *Text. Res. J.* **2023**, *93*, 1485–1503. [CrossRef]
7. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Proceedings, Part I 14; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37.
8. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
9. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
10. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
11. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
12. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1916. [CrossRef]
13. Zhao, J.; Zhou, S.; Zheng, Q.; Mei, S. Fabric defect detection based on transfer learning and improved Faster R-CNN. *J. Eng. Fibers Fabr.* **2022**, *17*, 15589250221086647.
14. Zhang, J.; Jing, J.; Lu, P.; Song, S. Improved MobileNetV2-SSDLite for automatic fabric defect detection system based on cloud-edge computing. *Measurement* **2022**, *201*, 111665. [CrossRef]
15. Zhou, S.; Zhao, J.; Shi, Y.S.; Wang, Y.F.; Mei, S.Q. Research on improving YOLOv5s algorithm for fabric defect detection. *Int. J. Cloth. Sci. Technol.* **2023**, *35*, 88–106. [CrossRef]
16. Wu, Y.; Lou, L.; Wang, J. Cotton fabric defect detection based on K-SVD dictionary learning. *J. Nat. Fibers* **2022**, *19*, 10764–10779. [CrossRef]
17. Lin, G.; Liu, K.; Xia, X.; Yan, R. An efficient and intelligent detection method for fabric defects based on improved YOLOv5. *Sensors* **2022**, *23*, 97. [CrossRef]
18. Guo, Y.; Kang, X.; Li, J.; Yang, Y. Automatic Fabric Defect Detection Method Using AC-YOLOv5. *Electronics* **2023**, *12*, 2950. [CrossRef]
19. Di, L.; Deng, S.; Liang, J.; Liu, H. Context receptive field and adaptive feature fusion for fabric defect detection. *Soft Comput.* **2023**, *27*, 13421–13434. [CrossRef]
20. Wang, C.Y.; Bochkovskiy, A.; Liao, H.Y.M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 7464–7475.
21. Liu, S.; Huang, D. Receptive field block net for accurate and fast object detection. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 385–400.
22. Baranwal, N.; Singh, K.N.; Singh, A.K. YOLO-based ROI selection for joint encryption and compression of medical images with reconstruction through super-resolution network. *Future Gener. Comput. Syst.* **2024**, *150*, 1–9.
23. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H. Designing network design strategies through gradient path analysis. *arXiv* **2022**, arXiv:2211.04800.
24. Zhu, L.; Lee, F.; Cai, J.; Yu, H.; Chen, Q. An improved feature pyramid network for object detection. *Neurocomputing* **2022**, *483*, 127–139. [CrossRef]
25. Chen, J.; Kao, S.h.; He, H.; Zhuo, W.; Wen, S.; Lee, C.H.; Chan, S.H.G. Run, Don't Walk: Chasing Higher FLOPS for Faster Neural Networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 17–24 June 2023; pp. 12021–12031.
26. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 116–131.
27. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Wang, W.; Zhu, Y.; Pang, R.; Vasudevan, V.; et al. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 1314–1324.
28. Tang, Y.; Han, K.; Guo, J.; Xu, C.; Xu, C.; Wang, Y. GhostNetv2: Enhance cheap operation with long-range attention. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 9969–9982.

29. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [CrossRef]
30. Wang, C.Y.; Liao, H.Y.M.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W.; Yeh, I.H. CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 13–19 June 2020; pp. 390–391.
31. Ding, P.; Qian, H.; Bao, J.; Zhou, Y.; Yan, S. L-YOLOv4: Lightweight YOLOv4 based on modified RFB-s and depthwise separable convolution for multi-target detection in complex scenes. *J. Real-Time Image Process.* **2023**, *20*, 71. [CrossRef]
32. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; Volume 31.
33. Jiang, N.; Wang, L. Quantum image scaling using nearest neighbor interpolation. *Quantum Inf. Process.* **2015**, *14*, 1559–1571. [CrossRef]
34. Manzhos, S.; Ihara, M. Neural network with optimal neuron activation functions based on additive Gaussian process regression. *arXiv* **2023**, arXiv:2301.05567.
35. Polyzos, E.; Nikolaou, C.; Polyzos, D.; Van Hemelrijck, D.; Pyl, L. Direct modeling of the elastic properties of single 3D printed composite filaments using X-ray computed tomography images segmented by neural networks. *Addit. Manuf.* **2023**, *76*, 103786. [CrossRef]
36. Siddique, A.; Vai, M.I.; Pun, S.H. A low cost neuromorphic learning engine based on a high performance supervised SNN learning algorithm. *Sci. Rep.* **2023**, *13*, 6280. [CrossRef] [PubMed]
37. Zheng, Z.; Wang, P.; Liu, W.; Li, J.; Ye, R.; Ren, D. Distance-IoU loss: Faster and better learning for bounding box regression. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12993–13000.
38. Sravya, N.; Lal, S.; Nalini, J.; Reddy, C.S.; Dell'Acqua, F. DPPNet: An efficient and robust deep learning network for land cover segmentation from high-resolution satellite images. *IEEE Trans. Emerg. Top. Comput. Intell.* **2022**, *7*, 128–139.
39. Tong, Z.; Chen, Y.; Xu, Z.; Yu, R. Wise-IoU: Bounding Box Regression Loss with Dynamic Focusing Mechanism. *arXiv* **2023**, arXiv:2301.10051.
40. Chen, M.; Yu, L.; Zhi, C.; Sun, R.; Zhu, S.; Gao, Z.; Ke, Z.; Zhu, M.; Zhang, Y. Improved faster R-CNN for fabric defect detection based on Gabor filter with Genetic Algorithm optimization. *Comput. Ind.* **2022**, *134*, 103551. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

applied sciences

MDPI

Article

A High-Quality Hybrid Mapping Model Based on Averaging Dense Sampling Parameters

Fanxiao Yi †, Weishi Li †, Mengjie Huang, Yingchang Du and Lei Ye *

College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China; yifanxiao@nudt.edu.cn (F.Y.); liweishi1028@nudt.edu.cn (W.L.); huangmj@nudt.edu.cn (M.H.); duyingchang@nudt.edu.cn (Y.D.)

* Correspondence: yelei@nudt.edu.cn

† These authors contributed equally to this work.

Abstract: Navigation map generation based on remote sensing images is crucial in fields such as autonomous driving and geographic surveying. Style transfer is an effective method for obtaining a navigation map of the current environment. However, there is lack of robustness of the map-style transfer model, resulting in unsatisfactory quality of the generated navigation maps. To address these challenges, we average the parameters of generators sampled from different iterations with a dense sampling strategy in the Generative Adversarial Network (CycleGAN). The results demonstrate that the training efficiency of our method on the MNIST and generation quality on the Google Map dataset are significantly superior to traditional style transfer methods. Moreover, it performs well in multi-environment hybrid mapping. Our method improves the generalization ability of the model and converts existing navigation maps to other styles of maps precisely. It can better adapt to different types of urban layout and road planning, bringing innovative solutions for traffic management and navigation systems.

Keywords: style transfer; remote sensing; model fusion; machine learning; deep learning

1. Introduction

Navigation maps [1–3], serving as a source of data, become more indispensable in fields such as autonomous driving [4], as more and more researchers use them for planning [5] and navigation decisions [6] in the context of autonomous vehicles. Point cloud data [7] and navigation systems [8] could acquire navigation maps, but they are time consuming, inefficient, and costly. In contrast, methods that use style transfer based on remote sensing images [9–11] to generate navigation maps are cost effective and offers real-time capability, which effectively address the aforementioned shortcomings. Style transfer [12,13] is one of the most significant parts of navigation map generation, which transforms images from one domain to another by using a style image to convert a given image without altering its original content. It extracts features from the content and style of the image separately, adjusting and optimizing the weights of neural networks, and then fuses them to reconstruct the final image. Currently, navigation maps are mainly used in urban environments and in off-road settings. Especially in some scenarios like autonomous vehicle testing, the generation of navigation maps based on remote sensing images can provide valuable off-road image samples and accurate geographic information and real-time updates [14–16].

Nevertheless, style transfer faces challenges such as insufficient precision and high time cost. In order to overcome the obstacles, we propose an improvement in optimization methods, which fuses the parameters of generators sampled from different iterations in CycleGANs [17]. The weight average (WA) of deep learning models leverages the strengths of different models to compensate for individual model deficiencies, giving rise to improved predictive outcomes [18,19]. However, the effect of CycleGAN combined

with WA is often not ideal. Considering that WA based on Dense Stochastic Weighted averaging (SWAD) [20] increases the diversity of samples and introduces cycle consistency constraints, we also average the weights of generators with a similar dense sampling strategy. The method improves the overall quality of style transfer and generalizability in off-road environments, which provides users with the best route planning and traffic information. In addition, it can also improve traffic efficiency and develop urban planning and future transportation.

Specifically, we improve the performance of CycleGAN by averaging the weights of generators sampled from different periods as shown in Figure 1. Image A is input into the network and transformed into *GeneratedB* (image B) by G_{A2B} (generators). Then, we establish a loss function between *Generated B* and image B via D_B (discriminators). *Cyclic A* is generated in reverse using G_{B2A} to establish the cycle loss, so as to maintain content consistency between *Generated B* and A. The SWAD is implemented within G_{A2B} and G_{B2A} by calculating weighted averages of model weights with the dense sampling strategy at different training stages. The method enables the model to efficiently extract features from off-road environments, improving its ability to generate navigation maps that blend both off-road and urban environments. Since each GAN model may have different training data and network architectures, the fusion process is able to mitigate the limitations of individual models and enhance the overall system's robustness, which makes it more adaptable to various input conditions [21,22].

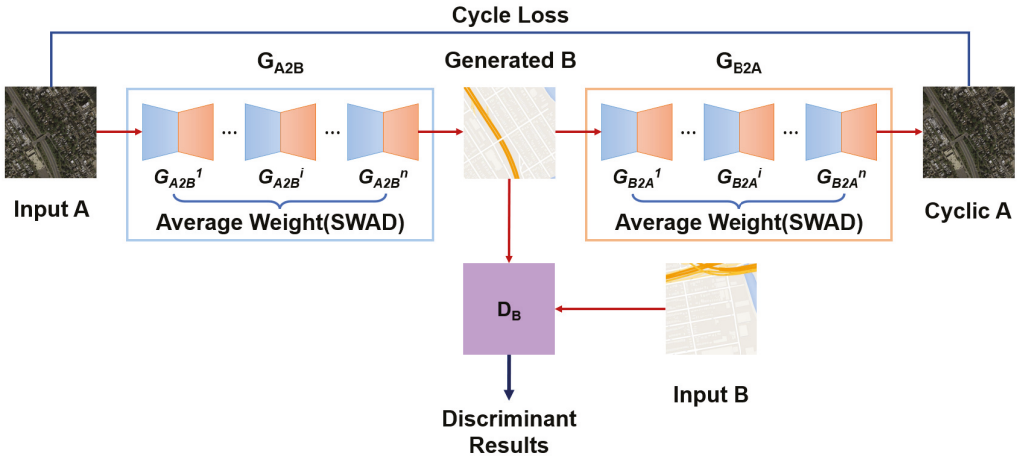


Figure 1. SWAD-CycleGAN fusion model network structure diagram. A denotes the A-class-style images including map information. *Generated B* denotes the final map with B-class-style images obtained by generators from A-style map information. *CyclicA* denotes the final map with A-class-style images obtained by generators from *Generated B*. G_{A2B} represents the generator that converts A-class-style images into B-class-style images, G_{B2A} represents the generator that converts B-class-style images into A-class-style images, and D_B represents the discriminator for B-class images. ${G_{A2B}}^i$ represents the model at round i in training.

The contributions made by this paper are as follows: we propose a novel method that averages the parameters of generators sampled from different iterations with the dense sampling strategy in CycleGANs. We conduct a large number of experiments to compare the performance of the method on datasets from Google Maps [23]. Additionally, we conduct separate training for urban and off-road environments. We confirm that the method not only conducts the convergence at a more rapid speed, but it also ensures the generation quality in both scenarios, ultimately resulting in the development of a hybrid-map-style transfer model. The remaining structure of this paper is organized as follows: Section 2 provides an overview of related work on GANs and the optimization methods employed. Section 3 gives priority to the mathematical formulation of the constructed models and the optimization methods adopted in our experiments. In Section 4, we present

comparative experiments and draw conclusions based on the results. Finally, we conclude our proposed method and offer prospects for future research.

2. Related Work

The work of map-style transfer focuses on transferring texture features in the early stage. However it does not explicitly address the style transfer task itself, resulting in less-than-ideal outcomes. With the development of deep learning technologies, some work [24] trains images with neural networks to make the image's style closer to the target style. But, these methods are computationally intensive and not suitable for real-time map generation, which tends to distortions in certain image data during transfer. Gatys et al. [25] simplify the style transfer problem as an optimization problem within a single neural network, of which the texture model based on deep image models was able to generate new images by matching feature representations of example images via an image search. However, it is not well suited for photos. Subsequently, Johnson et al. [26] expand on Gatys' work by replacing the input random noise image with a target image and incorporating a forward autoencoder network to model the style transfer process. Luan et al. [27] address the limitations of Gatys' model by constraining the output within a specific color space to suppress deformations and maintain the realism of photos. The constraint is fully differentiable, enabling better preservation of style information from reference images without introducing distortions caused by deformations. Li et al. [28] use feature transformations such as Whitening and Coloring Transform (WCT) to directly match the statistical properties of content feature statistics with those of style images in the deep feature space. Li et al. [29] improve the WCT-based method by introducing stylization and smoothing steps to tackle structural artifacts in the output of the WCT algorithm. However, the WCT-based approach still have the potential to produce spatial distortions or unrealistic artifacts. Yoo et al. [30] suggest incorporating wavelet transform into neural networks based on WCT, enabling features to retain both structural information and statistical properties of the VGG space, thereby addressing the aforementioned limitations. Recent work tends to use GANs to implement map-style transfer, especially CycleGAN [31]. However, there are challenges pertaining to image accuracy and real-time processing, which motivates us to enhance the CycleGAN for map-style transfer.

WA is a technique which fuses the deep learning models and aims to improve the model. Some recent work [32] uses Stochastic Weighted Averaging (SWA) [33] in CycleGANs [34,35], which enhances the capabilities of generators. Inspired by checkpoint averaging [36,37], Izmailov et al. [33] propose SWA. Using the mean of multiple models on the training trajectory can help find a better-performing network with no additional computational overhead compared to training a single model. When SGD convergence is good, the checkpoint of SWA sampling is concentrated around the optimal solution. Using SWA operations for these checkpoints helps optimize the training network. Similar to FGE [38–40], using the cyclic learning rate [33,41], models that are spatially close to each other but produce different predictions can be collected. Some recent work proposes variations on SWA that allow SWA to be adapted to more complex application scenarios and have more efficient rates. We tend to combine SWAD and SWA with CycleGANs to improve the performance of the navigation maps.

3. Methods

3.1. Cyclic Adversarial Generation Network Based on SWAD Optimization Method

CycleGAN realizes mapping from one domain to another via generators and discriminators. In this section, we propose a method that averages the parameters of generators sampled from different iterations with the dense sampling strategy in CycleGANs. First, we illustrate the framework of SWAD-CycleGAN, as shown in Figure 2. CycleGAN involves four models: G_{A2B}, G_{B2A}, D_A and D_B. G_{A2B} transforms *Real A* inputs into *Cyclic B* outputs, while G_{B2A} handles the transformation of *Cyclic B* inputs into *Cyclic A* outputs. The concept of cycle consistency in adversarial networks involves both forward cycle consistency

and backward cycle consistency. Forward cycle consistency refers to the fact that when an input from the source domain passes through a generator and a reverse generator, it should map back to an output within the source domain while maintaining consistency with the original input. Similarly, backward cycle consistency ensures that an input from the target domain, after going through the reverse generator and generator, should map back to an output within the target domain, remaining consistent with the original input. The cycle consistency loss helps in preventing the adversarial networks from learning mapping functions G and F, while learning mapping functions G and F tends to make the source and target domains have the same distribution. Despite the network having the capacity to map source domain images to any image in the target domain and generate a distribution equivalent to that of the target domain, cycle consistency loss is beneficial to constraining this behavior. Additionally, the identity loss function ensures that when A-class-style maps are inputted into G_{A2B}, the generated output resembles B-class-style images and vice versa. However, this may not always hold true, so the identity loss function is designed to impose constraints on this aspect.

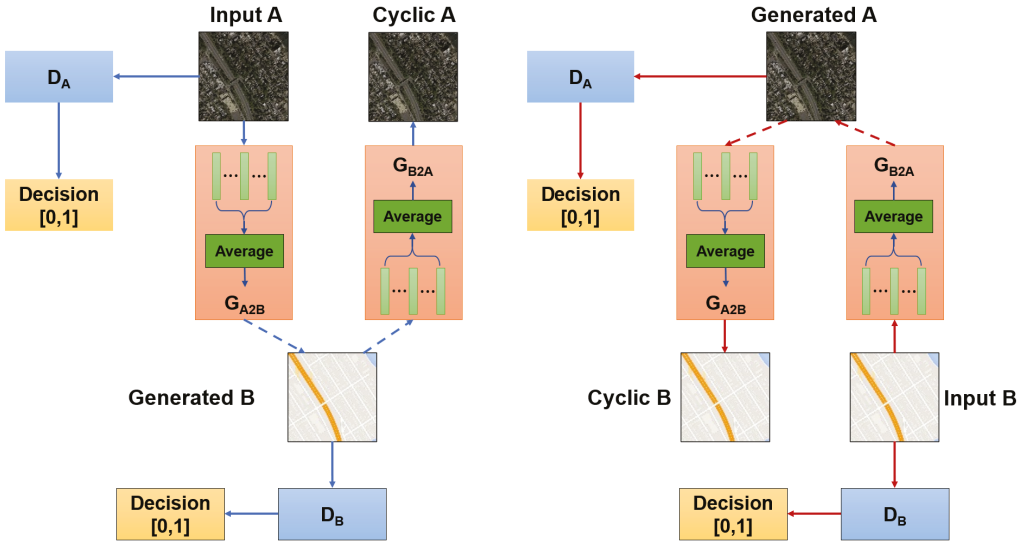


Figure 2. Forward experimental model and reverse experimental model of SWAD-CycleGAN.

D_A and D_B are responsible for discriminating between the authenticity of A and B. Additionally, there are four types of loss functions: G_{loss}, D_{loss}, $Cycle_{loss}$, and $identity_{loss}$. The loss function G_loss for the Generator as Equation (1), with a target to make the generated images closely align with the distribution of target images, ultimately "deceiving" the discriminator.

$$\mathcal{L}_{\mathrm{GAN}}(G, D_Y, X, Y) = \mathbb{E}_{y \sim p_{\mathrm{data}}(y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{\mathrm{data}}(x)}[\log(1 - D_Y(G(x))] \quad , \quad (1)$$

where $p_{data(y)}$ denotes probability distribution on the output. G and D denote the generators and discriminators. On the other hand, the loss function for the Discriminator (D_{loss}) has the opposite objective, namely the distinguishment of the images generated by the Generator. The cycle loss ($Cycle_{loss}$) function is depicted as Equation (2):

$$\mathcal{L}_{\mathrm{cyc}}(G, F) = \mathbb{E}_{x \sim p_{\mathrm{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y \sim p_{\mathrm{data}}(y)}[\|G(F(y)) - y\|_1]. \quad (2)$$

The generator and discriminator we designed are based on residual networks, and the specific internal network structures are shown in Figures 3 and 4. While previous work has mainly laid stress on improving loss functions, our target is to enhance the optimization methods for the four models and compare them with several state-of-the-art techniques.

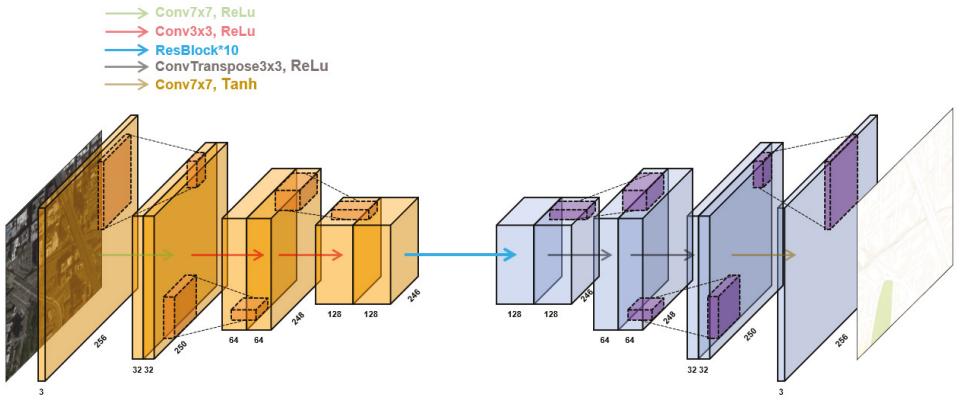


Figure 3. Neural network structure of the generators. ResBlock is a residual module composed of multiple convolutional layers and activation layers.

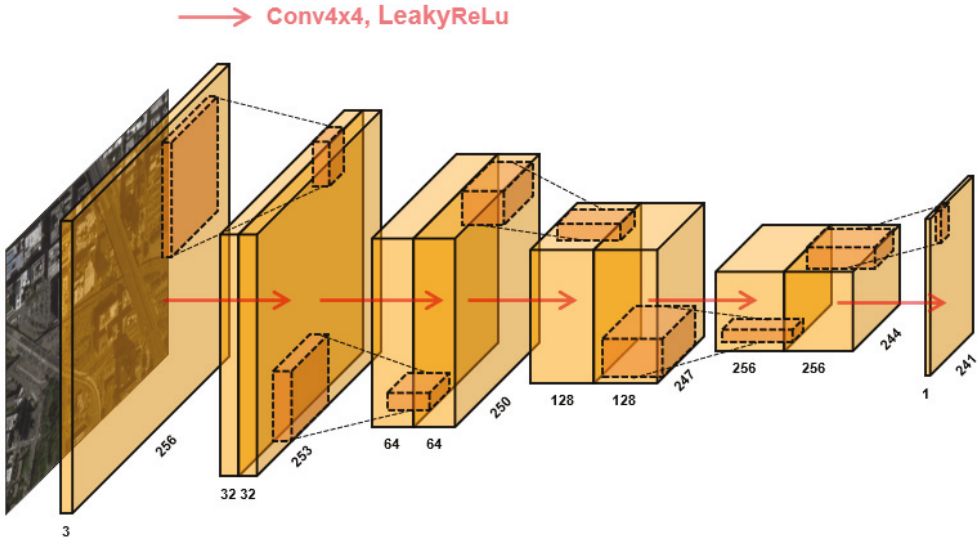


Figure 4. Neural network structure of the discriminators.

3.2. Network Optimization

Definitely, CycleGAN itself serves as a generative adversarial network for image transformation, which realizes mapping from one domain to another via two generators and two discriminators. We can fuse the output of two generators of CycleGAN to produce better results.

In this paper, we not only adopt SWAD to solve the instability and relieve mode oscillation problems in GAN model training, but also boost the robustness of the model. By averaging multiple best points in the training process, the model can better adapt to the unseen data. There are several generator networks in the GAN network, and through the SWAD strategy, the parameters of these generators can be averaged and the multiple generators can be "fused" into a unified generator model. While this reduces the difficulty in choosing the best generator, it also makes the model more stable and reliable. The SWA is based on averaging multiple points on the SGD trajectory with a periodic learning rate [33] as Equation (3):

$$\begin{aligned} \alpha(i) &= (1 - t(i))\alpha_1 + t(i)\alpha_2, \\ t(i) &= \frac{1}{c}(\mathrm{mod}\,(i-1, c) + 1). \end{aligned} \tag{3}$$

where α denotes the learning rate at i iteration, and c denotes the number of iterations. An improvement in generalization is accessible, only by averaging some points on the optimized trajectory during the training process. SWA significantly improves the training

of many state-of-the-art deep neural networks on a series of important baselines, with essentially no overhead. Considering that SWA averages the latter part of the network, the result cannot accurately approximate the ideal minimum on the high-dimensional parameter space. Therefore, we use SWAD [20] to gather enough weights in each iteration.

4. Experiments

The workstation configuration used for the experiments in this article consists of an AMD Ryzen 9 6900HX with 16 GB RAM. To evaluate the effectiveness of the SWAD method on general GAN networks, we first conduct tests on the MNIST [42] to ensure comprehensiveness. Subsequently, we evaluate the effectiveness and benefits of the SWAD-CycleGAN model on Google Maps [23]. Additionally, we conduct experiments in both urban and off-road environments to assess the generalization ability of SWAD-CycleGAN. The experimental results for these two environments are presented in this section.

We randomly split the data in each domain into two splits: 20% for an independent test set, and the remaining 80% for 10 times 10-fold cross validation to set hyperparameters. After that, the training set and validation set are combined and retrained to obtain the final model. Finally, we use the test set to assess the model's generalization ability.

4.1. Dataset and Hyperparameters

In the integrity testing phase of this study, we use the MNIST dataset which is composed of numbers handwritten by 250 different people, as shown in Table 1. During the experimental phase, we utilize RS maps and navigation maps provided by Google Maps which were sampled from in and around New York City. The Google Maps datasets also include urban and off-road environmental data from various locations, categorized into RS maps (class-A-style data) and navigation maps (class-B-style data), each with its respective training and testing subsets.

Table 1. The introduction of datasets.

Phase	Source	Image Size	Structure
Integrity Testing	MNIST	28×28	train test
Experiment	Google Maps	256×256	train: A class, B class test: A class, B class

The foundational neural network architecture used in this paper is the Residual Network [43] (ResNet). Consequently, the input size for the data was set at 256×256. For the training set, in order to ensure comprehensive training, the original data was initially magnified by a factor of 1.12 and subsequently randomly cropped to 256×256 dimensions. The data images were then subjected to random flips and finally normalized. Conversely, there were no such constraints applied to the testing set. The data was randomly split into a test set comprising 20% of the total data. The remaining 80% was divided into 10 parts. In each iteration, we randomly select nine parts as the training set, and the remaining part was used as the validation set, which ensures that each part was used as the validation set in turn. We repeat the process 10 times and average the results. The hyperparameters were adjusted based on the final results, as shown in Table 2.

Table 2. Hyperparameters employed in SWAD and traditional methods.

Hyperparameters	SGD [1]	Adam [1]	SWAD [1]	SWA [2]	PSWA [2]	SGD [2]	SWAD [2]
Learning rate	-	-	1×10^{-3}	1×10^{-4}	2×10^{-4}	2×10^{-4}	2×10^{-4}
Batch size	-	-	64	2	2	2	2
Number of epochs	-	-	100	100	100	100	100

[1] Integrity testing phase. [2] Experiment phase.

4.2. Integrity Testing of the SWAD Method on the GAN Network

Prior to the formal experiments, we conduct integrity testing on the SWAD optimization method to assess its feasibility and reliability. The integrity testing is performed using the MNIST datasets. We compare the performance of two conventional optimization methods, Stochastic Gradient Descent [44] (SGD) and Adaptive Moment Estimation [45] (Adam), with that of SWAD on the MNIST train set. Figure 5 presents the discriminator and generator loss based on three different optimization methods: Adam, SGD, and SWAD. Generated data is displayed every 10 epochs, with 16 samples randomly selected from the generated datasets. Over the course of 100 epochs, the SWAD method effectively reduces loss fluctuations, resulting in a steady decline. The SWAD method achieves a significantly lower discriminator loss and a slight decrease in generator loss, which performs exceptionally well on a GAN network. Generated data is displayed every 10 epochs, with 16 samples randomly selected from the generated datasets. Moreover, Figure 6 illustrates that by the 40th epoch, discernible handwritten numbers can already be observed. This suggests that the SWAD method rapidly converges and produces clear images, greatly improving training efficiency and image quality. In summary, the SWAD optimization method demonstrates faster convergence and a better ability to identify the optimal solution in gradient descent. This results in shorter training time while maintaining the quality of generated images. The method has successfully passed the integrity test and can be further applied to CycleGAN in a multi-environment in subsequent experiments.

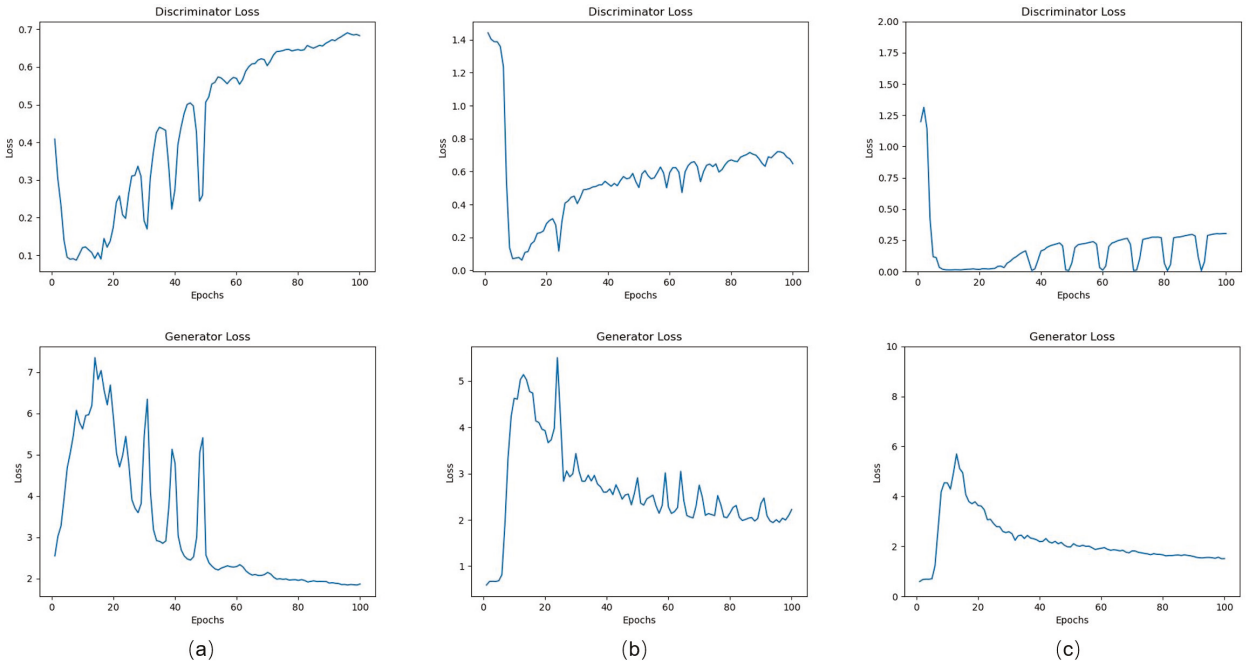


Figure 5. Loss reduction functions of various optimization methods under integrity testing. (**a**) The discriminator and generator loss of Adam. (**b**) The discriminator and generator loss of SGD. (**c**) The discriminator and generator loss of SWAD.

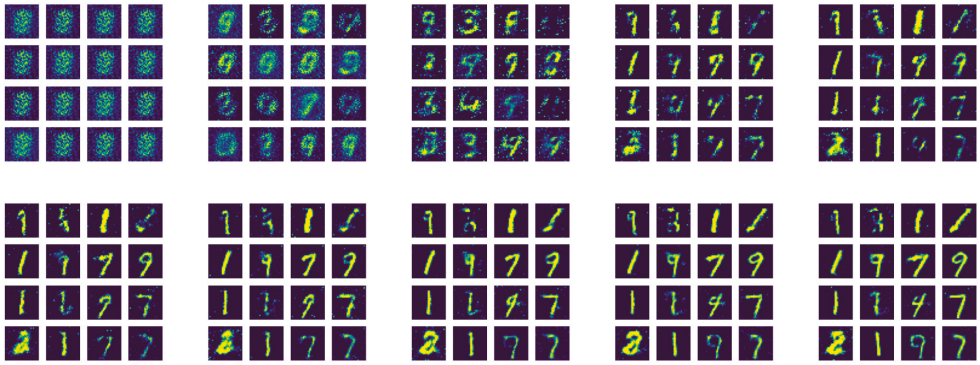


Figure 6. The generation results of SWAD on MNIST after 100 rounds of training (results sampled every 10 rounds).

4.3. Evaluation Indicators

The performance of the GAN's generators is evaluated based on the ability to deceive the discriminator with generated "fake maps". This iterative process aims to achieve a "Nash equilibrium", which is a non-cooperative game equilibrium between the generator and discriminator. To explore this, we conduct experiments using a "controlled variable method" where optimization algorithms are applied separately to the two types of models mentioned above, as well as simultaneously to both. The results indicate that applying the optimization algorithm to both models does not significantly improve their performance.

The quantitative indexes of navigation map generation quality include positioning accuracy, map coverage rate, update speed, path planning accuracy, real-time traffic information accuracy, etc. Generally, FID (Frechet inception distance score) [46] and IS (inception score) [47] are the main indicators used to evaluate the image quality generated by GANs. FID calculates the distance between feature vectors of real and generated images. A lower FID score indicates a closer similarity between the two images or more similar distributions. In the best case scenario, the FID score is 0.0, which represents identical images. Another indicator used is the IS (inception score) [47], which can be understood as a measure of image clarity or resolution during training. Since the primary goal of the generative model is to produce realistic images, the IS score reflects the clarity of the generated images. For a clear image, the values of vector y in one dimension are larger, indicating a higher probability of belonging to a certain class, while values in other dimensions are smaller. In professional terms, a clear image has lower entropy in $P(y \mid x)$ (higher entropy indicates greater chaos or uncertainty in the values of a random variable). The IS indicator is shown as Equation (4):

$$IS(G) = \mathbb{E}[Ex \sim PgKL(P(y \mid x)P(y))]. \tag{4}$$

In brief, the larger the IS index of the generated image, the clearer and higher the resolution, which is obviously expected by the generated model.

4.4. Experiment Results

4.4.1. Experiment Results of Prevailing and Proposed Methods on Google Maps Dataset

In the initial phase, we conduct 200 rounds of training. The training results show that after approximately 100 rounds, all indicators display minimal fluctuations, indicating convergence. Therefore, we present the results and various parameter metrics after 100 rounds of training. To ensure the completeness of the model, we conduct forward and reverse experiments for each set of methods. This means that the model should be able to transform from satellite map-style class-A data to navigation map-style class-B-style data and vice versa. Table 1 presents the parameter values for the various methods used in the four comparative experiments. IS-B refers to the inception score of the generated class-B-style data in the forward process (satellite map to navigation map). IS-A refers to

the inception score of the generated class-A-style data in the reverse process (navigation map to satellite map).

First, from Table 3, the combination of CycleGAN with SWAD performs best than SWA and other methods. Compared to SWA, when SWAD is deployed on the generators, the FID indicator decreases by 55.8% and the $IS\text{-}B$ indicator has increased by 28.9%. It means that the gap of SWAD between the generated data and real data decreases significantly compared to SWA or SGD. At the same time, the quality of both A-class-style and B-class-style generated data improves considerably, which is a favorable outcome for GANs. Moreover, when PSWA [37] is deployed on both the generators, it improves the $IS\text{-}A$ indicator, but it performs less well on other indicators such as FID.

Table 3. The performance indicators based on prevailing and proposed methods on test dataset.

Methods	*FID*	Decreasing Rate of *FID* [1]	*IS-A*	*IS-B*
SWA	194.970	0	4.328	2.334
SWA (generators and discriminators) [2]	331.962	−70.3%	3.897	2.097
PSWA	246.689	−26.5%	4.756	1.796
SGD	89.147	54.3%	3.696	3.004
SWAD	86.274	55.8%	3.892	3.008

[1] Using SWA (only generators) as a reference. [2] Other methods only use SWA for generators by default.

Second, the method works better when deployed on generators than deployed on both generators and discriminators. When SWAD is deployed only on the generators, the gap between generated data and real data decreases significantly compared to the previous experiments. Although it is still larger than the baseline, the quality of generated A-class data is even higher, and the quality of generated B-class data also improves considerably. This indicates that deploying SWAD on the generator alone yields better results than deploying it simultaneously. When SWAD is simultaneously deployed on both the generator and discriminator, it amplifies the gap between generated data and real data. This is because the model inherently involves an adversarial iterative process, where the generator and discriminator compete with each other to achieve training effectiveness. Deploying the optimization method on both sides inevitably leads to one side being weaker and the other stronger, which is clearly not conducive to training. Hence, the elevated value of this indicator is understandable.

Third, compared to SGD, the indicators of SWA related to the quality of generation are better, which indicates that SWDA improves the resolution of generated images. Additionally, the effect of SWAD on FID, $IS\text{-}A$, and $IS\text{-}B$ is higher than SGD, but other SWA variants still fall short of the SGD indicators. This further highlights that the network's adversarial performance is not solely dependent on the optimization method.

Figure 7 demonstrates the loss reduction process of various methods during the training process. Each column corresponds to a different optimization method. Within each column, the top section represents the loss when discriminating A-class images (D_A), the middle section represents the loss when discriminating B-class-style images (D_B), and the bottom section represents the total loss of the generator. To compare the convergence speed and oscillation level of different optimization methods during training, we perform statistical analysis on the total loss of generator shown at the bottom of Figure 7 and document the results in Table 4. In order to standardize the convergence criteria, we establish the following guidelines:

1. Convergence is considered when the amplitude of the loss curve does not exceed 10% of the maximum loss value, and the training round at this point is identified as the convergence time.
2. We record the oscillation counts equal to or greater than 10% of the maximum loss value to assess the extent of oscillation during training.

Table 4. The convergence rounds and oscillation counts for each optimization method.

Methods	Convergence Rounds	Oscillation Counts
SWA	42 k	40
SWA (generators and discriminators)	35 k	35
PSWA	48 k	51
SGD	36 k	46
SWAD	26 k	36

As shown in Table 4, the convergence speed of SWA is nearly identical to that of SGD, but it significantly reduces the degree of oscillation. On the other hand, the PSWA method not only converges slower but also intensifies the oscillation. In contrast, SWAD demonstrates better performance by significantly accelerating the convergence speed and reducing the degree of oscillation, leading to a more stable training process.

Figure 7 indicates that the SWA is more effective in identifying the optimal solution compared to the baseline, enabling it to reach the lowest gradient more rapidly. However, this also leads to a higher generator loss for the same batch compared to the baseline. In panel (d), both the generator and discriminator exhibit a more balanced performance, ultimately surpassing the baseline. The experimental results further demonstrate that the SWA optimization method effectively addresses the two longstanding issues encountered in map-style transfer tasks: time-consuming processes and poor quality.

The total loss of the generator consists of three components: $G_{GAN_{loss}}$ (composed solely of the loss from the G_{A2B} and G_{B2A} generators), $G_{cycle_{loss}}$, and $G_{identity_{loss}}$. The loss functions for these components are depicted in Figure 8.

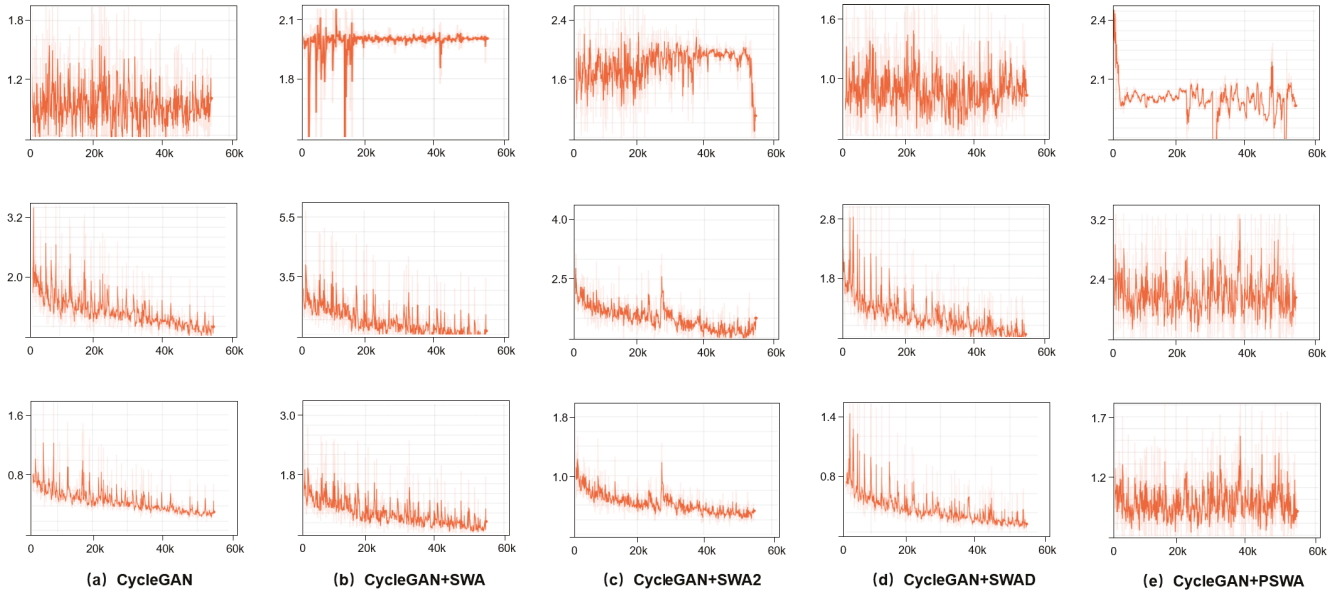


Figure 7. Training loss for each optimization method. (**Top**) loss of discriminator *A*; (**Middle**) loss of discriminator *B*; and (**Bottom**) total loss of generator.

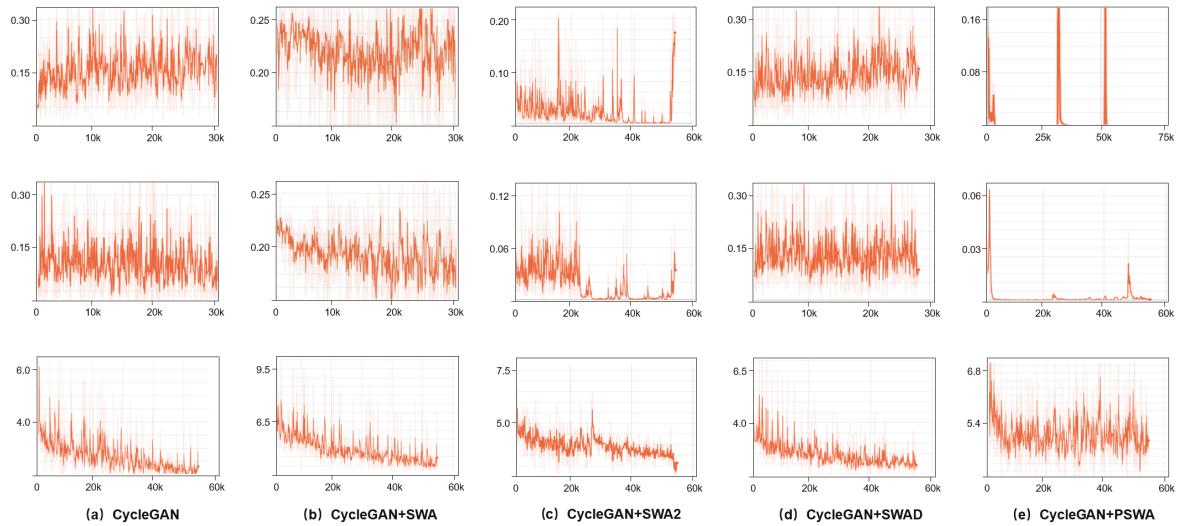


Figure 8. Three components of the total loss in the generator. (**Top**) GAN loss; (**Middle**) cycle loss; and (**Bottom**) identity loss.

4.4.2. Style Transfer Result on Google Maps Dataset

We select urban and off-road environment datasets for group training. After completing the training, we input class-A-style data or class-B-style data into the model to generate the corresponding transformed image. In the following sections, we display and analyze the output results of the PSWA and SWAD methods. These results consist of 16 randomly selected outputs from over a thousand results obtained in urban and off-road environments. The relationship between the input and output is as follows: when the class-A-style test set is inputted, a class-B-style converted image is generated, and when the class-B-style test set is inputted, a class-A-style converted image is generated. Figures 9 and 10 illustrate the forward migration process using the PSWA method, while Figure 11 depicts the reverse migration process using the PSWA method. Additionally, Figures 12 and 13 demonstrate the forward migration process using the SWAD method, and Figure 14 showcases the reverse migration process using the SWAD method.



Figure 9. The forward transfer process in urban environments based on PSWA.

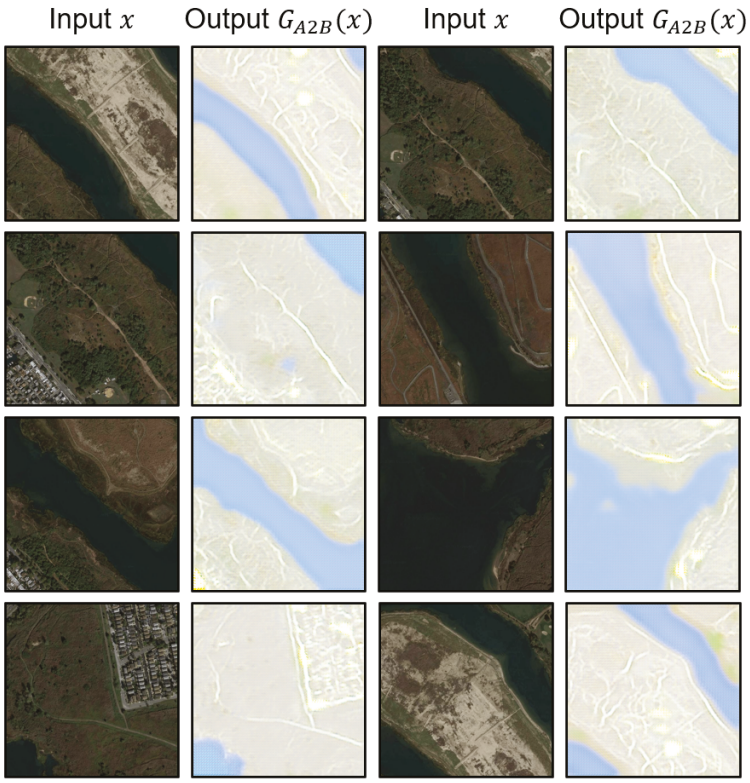


Figure 10. The forward transfer process in off-road environments based on PSWA.

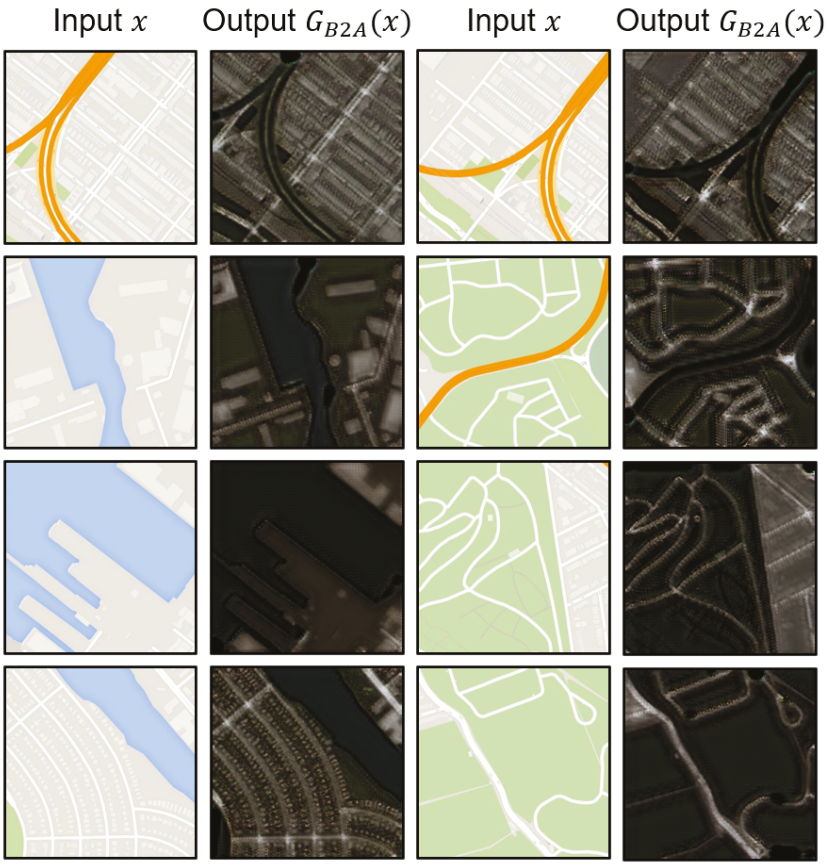


Figure 11. The reverse transfer process based on PSWA.

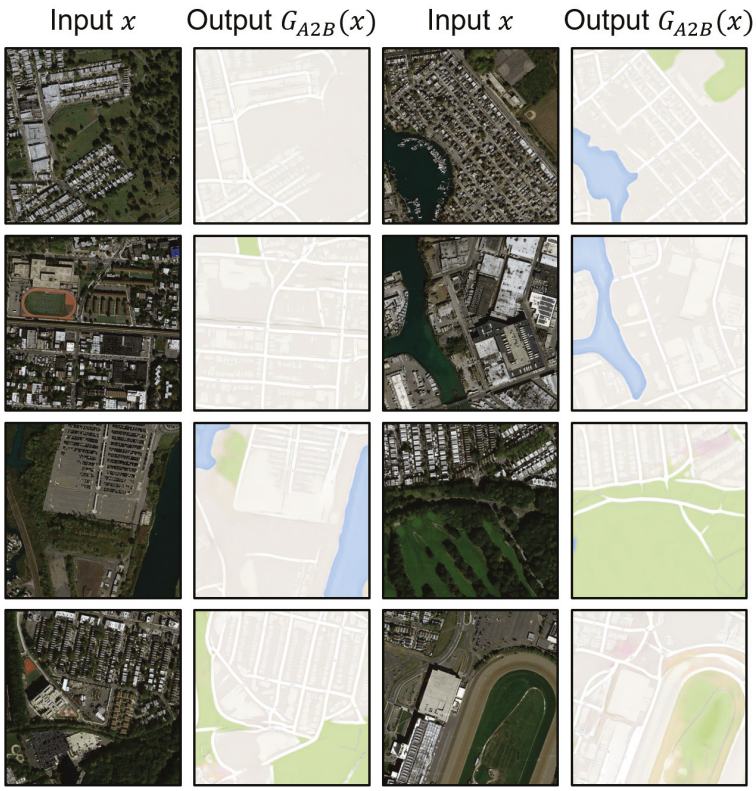


Figure 12. The forward transfer process in urban environments based on SWAD.

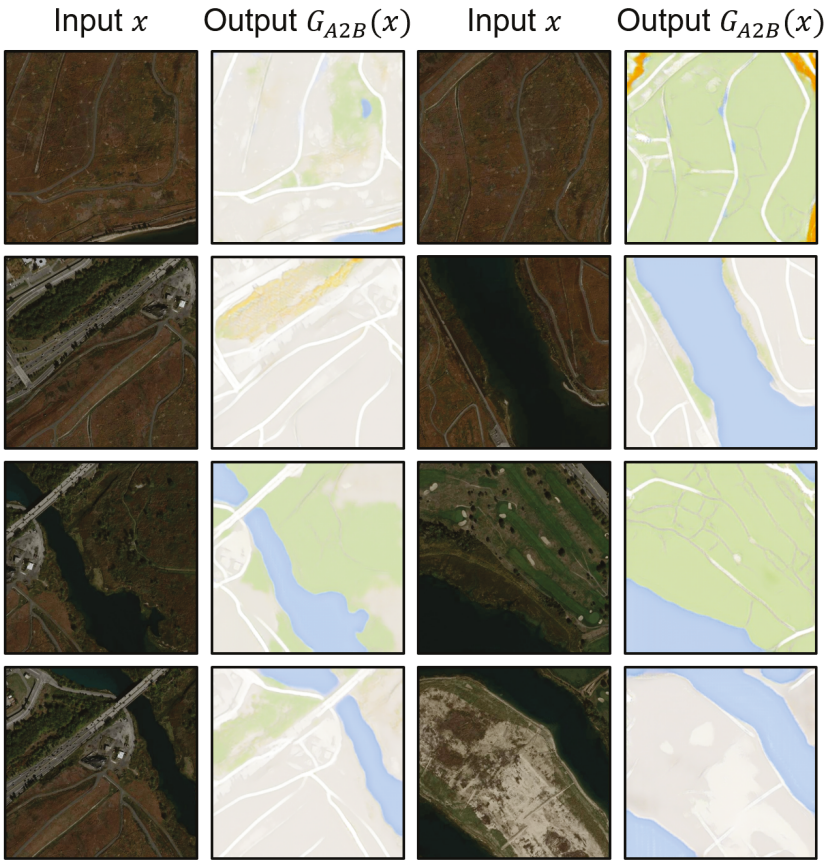


Figure 13. The forward transfer process in off-road environments based on SWAD.



Figure 14. The reverse transfer process based on SWAD.

As observed in Figures 9 and 10, it is apparent that the forward migration effect of PSWA is not satisfactory. The map outline drawn appears blurred, and the migration effect of certain key elements is subpar. Overall, this can be attributed to the fact that although PSWA partially addresses the issue of over-fitting in ordinary SWA, it still lacks sufficient training, resulting in a decrease in the clarity of the generated images. Furthermore, the data adaptability to off-road environments is limited, and the mapping accuracy is low, which does not meet the requirements for hybrid mapping in multiple environments. Nevertheless, it is evident that PSWA exhibits a strong resemblance to the source domain, thereby demonstrating its potential to enhance the model's robustness.

Figure 11 illustrates the reverse migration process of the PSWA method. The outlines of the buildings in the map appear to be clearer, although still somewhat blurry. The migration effect is more pronounced for data with significant differences in characteristics, but the distinction between the reverse generation of data with similar characteristics is low. Specifically, there is not a significant differentiation between rivers and green spaces in the data.

From Figures 12 and 13, we can observe that the SWAD method effectively addresses the issue of over fitting without compromising the quality of generation. Furthermore, the clarity of the SWAD method surpasses the generation effect of the PSWA method. In urban environments, distinct features such as vegetation, lakes, and rivers are adequately represented in the remote sensing images and exhibit good transfer results. Boundaries like roads and sidewalks are also well reflected in the navigation maps. In off-road environments, the elements in the remote sensing images are relatively uniform, mainly consisting of deserts or channels. As a result, the transfer effect is good, essentially achieving the goal of generating hybrid-transfer maps between urban and off-road environments.

When inputting class-B-style images, it should be possible to perform a reverse style transfer back to class-A-style images. Therefore, the results of the reverse transfer are shown in Figure 14. It can be observed that although some empty areas in the remote

sensing images can only be filled with houses or other objects generated by the algorithm after the transfer, the overall segmentation lines and block restoration are relatively good, resulting in a satisfactory outcome.

5. Conclusions

In this paper, we propose a novel method that averages the parameters of generators sampled from different iterations with the dense sampling strategy in CycleGANs. Two networks of generators converse images from one domain to another. The fusion of generators improves the conversion quality, increases sample diversity, balances generator strengths and weaknesses, and reduces problems such as mode crashes and oscillations. In future work, we will generalize the method under different datasets and scenarios to adapt to migration tasks in different environments. In sum, the fusion of GAN generators is an efficient method to further improve the performance of image conversion in CycleGAN, bringing more diverse and authentic results. However, the computing time and resource consumption will increase significantly in large-scale occasions. Also, the adjustment of parameters may increase the cost of training. The combination of WA and the generation of navigation maps based on style transfer has a potential future. With the development of deep learning and generative models, there may be a more efficient and accurate WA technique of GANs in the future, making navigation image generation more intelligent and realistic. In addition, combined with other technologies such as reinforcement learning and augmented reality, navigation image generation may further explore the fields of application and provide a more rich and interactive navigation experience.

Author Contributions: Conceptualization, L.Y. and F.Y.; methodology, F.Y. and W.L.; software, F.Y. and W.L.; validation, F.Y., W.L. and M.H.; formal analysis, F.Y. and W.L.; investigation, Y.D. and M.H.; resources, F.Y. and W.L.; data curation, F.Y.; writing—original draft preparation, F.Y. and W.L.; writing—review and editing, W.L.; visualization, F.Y. and W.L.; supervision, L.Y.; project administration, L.Y.; funding acquisition, L.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the author Fanxiao Yi (yi_fanxiao@foxmail.com). The data are not publicly available due to our laboratory's confidentiality agreement and policies.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Huang, C.; Mees, O.; Zeng, A.; Burgard, W. Visual language maps for robot navigation. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 10608–10615.
2. Huang, C.; Mees, O.; Zeng, A.; Burgard, W. Audio visual language maps for robot navigation. *arXiv* **2023**, arXiv:2303.07522.
3. Mao, J.H.; Yang, J.; Shao, R.P.; Wang, W.Z. Research on the construction of a BIM-based model for cross-floor indoor navigation maps. In *Frontiers in Civil and Hydraulic Engineering*; CRC Press: Boca Raton, FL, USA, 2023; Volume 1, pp. 372–378.
4. Grigorescu, S.; Trasnea, B.; Cocias, T.; Macesanu, G. A survey of deep learning techniques for autonomous driving. *J. Field Robot.* **2020**, *37*, 362–386. [CrossRef]
5. Jiang, Z.; Zhang, X.; Wang, P. Grid-Map-Based Path Planning and Task Assignment for Multi-Type AGVs in a Distribution Warehouse. *Mathematics* **2023**, *11*, 2802. [CrossRef]
6. Yamaguchi, T.; Kuwano, A.; Koyama, T.; Okamoto, J.; Suzuki, S.; Okuda, H.; Saito, T.; Masamune, K.; Muragaki, Y. Construction of brain area risk map for decision making using surgical navigation and motor evoked potential monitoring information. *Int. J. Comput. Assist. Radiol. Surg.* **2023**, *18*, 269–278. [CrossRef] [PubMed]
7. Zhang, Q.; Liu, X. Robot indoor navigation point cloud map generation algorithm based on visual sensing. *J. Intell. Syst.* **2023**, *32*, 20220258. [CrossRef]
8. Tanwar, J.; Sharma, S.K.; Mittal, M. Designing obstacle's map of an unknown place using autonomous drone navigation and web services. *Int. J. Pervasive Comput. Commun.* **2023**, *19*, 154–169. [CrossRef]

9. Ma, L.; Liu, Y.; Zhang, X.; Ye, Y.; Yin, G.; Johnson, B.A. Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS J. Photogramm. Remote Sens.* **2019**, *152*, 166–177. [CrossRef]
10. Salvo, C.; Vitale, A. A Remote Sensing Method to Assess the Future Multi-Hazard Exposure of Urban Areas. *Remote Sens.* **2023**, *15*, 4288. [CrossRef]
11. Wang, L.; Gao, R.; Li, C.; Wang, J.; Liu, Y.; Hu, J.; Li, B.; Qiao, H.; Feng, H.; Yue, J. Mapping Soybean Maturity and Biochemical Traits Using UAV-Based Hyperspectral Images. *Remote Sens.* **2023**, *15*, 4807. [CrossRef]
12. Jing, Y.; Yang, Y.; Feng, Z.; Ye, J.; Yu, Y.; Song, M. Neural style transfer: A review. *IEEE Trans. Vis. Comput. Graph.* **2019**, *26*, 3365–3385. [CrossRef]
13. Wang, P.; Li, Y.; Vasconcelos, N. Rethinking and improving the robustness of image style transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 124–133.
14. Ayyalasomayajula, R.; Arun, A.; Wu, C.; Sharma, S.; Sethi, A.R.; Vasisht, D.; Bharadia, D. Deep learning based wireless localization for indoor navigation. In Proceedings of the 26th Annual International Conference on Mobile Computing and Networking, London, UK, 21–25 September 2020; pp. 1–14.
15. Fernández, C.; Munoz-Bulnes, J.; Fernández-Llorca, D.; Parra, I.; Garcia-Daza, I.; Izquierdo, R.; Sotelo, M.A. High-level interpretation of urban road maps fusing deep learning-based pixelwise scene segmentation and digital navigation maps. *J. Adv. Transp.* **2018**, *2018*, 2096970. [CrossRef]
16. Golroudbari, A.A.; Sabour, M.H. Recent Advancements in Deep Learning Applications and Methods for Autonomous Navigation–A Comprehensive Review. *arXiv* **2023**, arXiv:2302.11089.
17. Lee, Y.W.; Kim, J.S.; Park, K.R. Ocular Biometrics with Low-Resolution Images Based on Ocular Super-Resolution CycleGAN. *Mathematics* **2022**, *10*, 3818. [CrossRef]
18. Xu, C.; Shu, J.; Zhu, G. Multi-Feature Dynamic Fusion Cross-Domain Scene Classification Model Based on Lie Group Space. *Remote Sens.* **2023**, *15*, 4790. [CrossRef]
19. Singh, S.P.; Jaggi, M. Model fusion via optimal transport. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 22045–22055.
20. Cha, J.; Chun, S.; Lee, K.; Cho, H.C.; Park, S.; Lee, Y.; Park, S. Swad: Domain generalization by seeking flat minima. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 22405–22418.
21. Li, J.; Hong, D.; Gao, L.; Yao, J.; Zheng, K.; Zhang, B.; Chanussot, J. Deep learning in multimodal remote sensing data fusion: A comprehensive review. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *112*, 102926. [CrossRef]
22. Ghassemian, H. A review of remote sensing image fusion methods. *Inf. Fusion* **2016**, *32*, 75–89. [CrossRef]
23. 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, Honolulu, HI, USA, 21–26 July 2017; pp. 1125–1134.
24. Gatys, L.; Ecker, A.S.; Bethge, M. Texture synthesis using convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 262–270.
25. 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, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 2414–2423.
26. Johnson, J.; Alahi, A.; Fei-Fei, L. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016*; Proceedings, Part II 14; Springer: Cham, Switzerland, 2016; pp. 694–711.
27. Luan, F.; Paris, S.; Shechtman, E.; Bala, K. Deep photo style transfer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4990–4998.
28. Li, Y.; Fang, C.; Yang, J.; Wang, Z.; Lu, X.; Yang, M.H. Universal style transfer via feature transforms. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 385–395.
29. 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), Munich, Germany, 8–14 September 2018; pp. 453–468.
30. Yoo, J.; Uh, Y.; Chun, S.; Kang, B.; Ha, J.W. Photorealistic style transfer via wavelet transforms. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 9036–9045.
31. 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, Venice, Italy, 22–29 October 2017; pp. 2223–2232.
32. Li, B.; Li, P.; Liu, B.; Li, M. A High-Precision Underwater Target Detection Method Based on Cascade Neural Network and Edge Computing. CN116758406A, 15 September 2023.
33. Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.; Wilson, A.G. Averaging weights leads to wider optima and better generalization. *arXiv* **2018**, arXiv:1803.05407.
34. Son, D.M.; Kwon, H.J.; Lee, S.H. Enhanced Night-to-Day Image Conversion Using CycleGAN-Based Base-Detail Paired Training. *Mathematics* **2023**, *11*, 3102. [CrossRef]
35. Krstanović, L.; Popović, B.; Janev, M.; Brkljač, B. Feature Map Regularized CycleGAN for Domain Transfer. *Mathematics* **2023**, *11*, 372. [CrossRef]
36. Chen, H.; Lundberg, S.; Lee, S.I. Checkpoint ensembles: Ensemble methods from a single training process. *arXiv* **2017**, arXiv:1710.03282.
37. Guo, H.; Jin, J.; Liu, B. Stochastic weight averaging revisited. *Appl. Sci.* **2023**, *13*, 2935. [CrossRef]

38. Garipov, T.; Izmailov, P.; Podoprikhin, D.; Vetrov, D.P.; Wilson, A.G. Loss surfaces, mode connectivity, and fast ensembling of dnns. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 8789–8798.
39. Huang, G.; Li, Y.; Pleiss, G.; Liu, Z.; Hopcroft, J.E.; Weinberger, K.Q. Snapshot ensembles: Train 1, get m for free. *arXiv* **2017**, arXiv:1704.00109.
40. Neklyudov, K.; Molchanov, D.; Ashukha, A.; Vetrov, D. Variance networks: When expectation does not meet your expectations. *arXiv* **2018**, arXiv:1803.03764.
41. Mandt, S.; Hoffman, M.D.; Blei, D.M. Stochastic gradient descent as approximate bayesian inference. *arXiv* **2017**, arXiv:1704.04289.
42. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
43. 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, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
44. Robbins, H.; Monro, S. A stochastic approximation method. *Ann. Math. Stat.* **1951**, *22*, 400–407. [CrossRef]
45. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
46. Obukhov, A.; Krasnyanskiy, M. Quality assessment method for GAN based on modified metrics inception score and Fréchet inception distance. In *Proceedings of the Software Engineering Perspectives in Intelligent Systems: Proceedings of 4th Computational Methods in Systems and Software 2020*; Springer: Cham, Switzerland, 2020; Volume 14, pp. 102–114.
47. Chong, M.J.; Forsyth, D. Effectively unbiased fid and inception score and where to find them. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6070–6079.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

MDPI AG
Grosspeteranlage 5
4052 Basel
Switzerland
Tel.: +41 61 683 77 34

Applied Sciences Editorial Office
E-mail: applsci@mdpi.com
www.mdpi.com/journal/applsci



Disclaimer/Publisher's Note: The title and front matter of this reprint are at the discretion of the Guest Editors. The publisher is not responsible for their content or any associated concerns. The statements, opinions and data contained in all individual articles are solely those of the individual Editors and contributors and not of MDPI. MDPI disclaims responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.





Academic Open
Access Publishing

mdpi.com

ISBN 978-3-7258-9124-5