Publisher: State and Provincial Joint Engineering Lab. of Advanced Network Monitoring and Control (ANMC)

Cooperate:

Xi'an Technological University (CHINA)
West Virginia University (USA)
Huddersfield University of UK (UK)
Missouri Western State University (USA)
James Cook University of Australia
National University of Singapore (Singapore)

Approval:

Library of Congress of the United States

Shaanxi provincial Bureau of press, Publication, Radio and Television

Address:

4525 Downs Drive, St. Joseph, MO64507, USA

No. 2 XueFu Road, WeiYang District, Xi’an, 710021, China

Telephone: +1-816-2715618 (USA) +86-29-86173290 (CHINA)

Website: www.ijanmc.org

E-mail: ijanmc@ijanmc.org

xxwlcn@163.com

ISSN: 2470-8038

Print No. (China): 61-94101

Publication Date: April 22, 2025

Editor in Chief

Associate Editor-in-Chief

Peter the Great St. Petersburg Polytechnic University, Russia

Administrator

Dr. & Prof. George Yang
Department of Engineering Technology
Missouri Western State University, St. Joseph, MO 64507, USA

Professor Zhongsheng Wang
Xi'an Technological University, China
State and Provincial Joint Engineering Lab. of Advanced Network and Monitoring Control, China

Associate Editors

Prof. Yuri Shebzukhov
International Relations Department, Belarusian State University of Transport, Republic of Belarus.

Dr. & Prof. Changyuan Yu
Dept. of Electrical and Computer Engineering, National Univ. of Singapore (NUS)

Dr. Omar Zia
Professor and Director of Graduate Program
Department of Electrical and Computer Engineering Technology
Southern Polytechnic State University
Marietta, Ga 30060, USA

Dr. Baolong Liu
School of Computer Science and Engineering
Xi'an Technological University, CHINA

Language Editor

Would you like to be an Associate Editor? Simply send a request together with your Curriculum Vitae to xxwlcn@163.com. We will have a team of existing editors or at least three experts in your field to review your request and make a decision as soon as we can. The criteria to be an associate editor are: 1. must have advanced degree; 2. must be a leader or have outstanding achievements in the specific research field; 3. must be recommended by the review team.

Table of Contents

Improved Method of ResNet50 Image Classification Based on Transfer Learning

Tao Shi
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: s19829601144@163.com

Zhiyi Hu
Engineering Design Institute
Army Research Loboratory
Beijing, 100042, China
E-mail: 763757335 @qq.com

Jun Yu
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: yujun@xatu.edu.cn

Kuncai Jiang
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: 2992165628@qq.com

***Abstract*—Aiming at the issues of high computational cost and limited generalization ability of ResNet50 in classifying images, this study advances an optimization strategy based on transfer learning. The model is initialized with transfer learning to reduce computational burden, and data augmentation techniques are employed to enhance generalization ability. Additionally, label smoothing is introduced to optimize the cross-entropy loss, thereby reducing sensitivity to noisy labels. The training process is further optimized using cosine annealing learning rate decay. Experimental findings reveal that the optimized ResNet50 model achieves a 6.25% improvement in classification accuracy on the CIFAR-10 dataset, validating the validity of the suggested methods.**

Keywords-Transfer Learning; ResNet50; Data Augmentation; Image Classification

I. INTRODUCTION

With the computer vision community's growing reliance on deep learning systems, CNNs have evolved into cornerstone solutions for visual classification problems [1-2]. Among them, ResNet (Residual Networks), as a deep residual network, has shown excellent performance in image classification tasks. Through its unique residual structure, it effectively addresses the issue of gradient vanishing and has achieved satisfactory results in many scenarios. However, in practical applications, the ResNet50 model still exposes some problems, such as large computational load, long training time, and a tendency to overfit [3-5]. Especially in efficient classification tasks, how to ensure classification accuracy while improving computational efficiency has become an urgent challenge. Jiang Zhengfeng et al. proposed combining the attention mechanism with deep residual networks for the categorization of remote sensing image scene data [6], achieving an accuracy of 92.94%. Zhang Yizhuo advanced a hierarchical fusion strategy rooted in residual architectures for hyperspectral ground object classification [7], achieving an average overall accuracy of 98.75%. Fang Liang et al. proposed a categorization approach for rusted steel bars rooted in deep residual networks, using industrial camera images combined with data enhancement techniques to classify the rust levels of six datasets [8], with classification accuracies all above 93.2%, and the highest reaching 98.8%. Although these methods have achieved good results in their respective application scenarios, they still have certain limitations when dealing with multi-classification problems.

DOI: 10.2478/ijanmc-2025-0011

Rooted in this, this study advances an approach to improve the performance of ResNet50 image classification rooted in transfer learning. This approach exploits the idea of transfer learning to optimize a priorly trained model, reducing the huge computational cost of training from scratch and accelerating network convergence. At the same time, data enhancement techniques are introduced to effectively solve the issues of excessive fitting and inadequate generalization capacity in model training; the label smoothing method is used to modify the cross-entropy loss function to alleviate the oscillation of the model loss value; the cosine annealing decay method is used to train the model, further accelerating the network convergence speed and improving grouping exactness. Verified on the CIFAR-10 data set, the image grouping exactness of this model reaches 93.75%, fully demonstrating its good classification ability.

II. RESNET NETWORK MODEL

ResNet (Residual Neural Network) was advanced by HE Kaiming. from Microsoft Research [9]. It serves as a key component in modern image classification systems. The proposal of ResNet's network structure has improved the training speed and model accuracy of neural networks. With deeper network architectures, the model's complexity grows significantly, and if the number of layers exceeds a reasonable range, the vanishing gradient issue may emerge. To solve this problem, scholars proposed the residual structure learning unit, as shown in Figure 1. This architecture aims to improve the approximation capability of the nonlinear stacked layers f(x) toward H(x) with increasing depth, where residual connections enable F(x)+x to approximate H(x) [10]. At the same time, by adding identity mapping in deep networks, shallow features can be directly transmitted to deep networks, avoiding the problem of gradient disappearance during backpropagation, thereby preventing network performance degradation and improving the performance of deep networks.

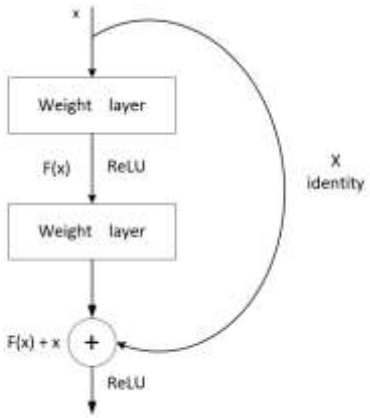


Figure 1. Residual unit structure diagram

As illustrated in Figure 1, the input x first passes through an initial weight layer to generate F(x). After processing by a ReLU activation function, it proceeds to a second weight layer. A skip connection directly adds the original input x to this output, producing the final mapping F(x)+x. This architectural design enables direct feature propagation from shallow to deep layers, facilitating inter-layer information flow.

ResNet series network models are commonly employed across multiple fields, with significant adoption in processing medical images and analyzing satellite imagery. ResNet50 is a typical model in the ResNet series, indicating that the network has 50 layers. This article employs ResNet50 as the foundational architecture for investigation, the architectural details are presented in Table Ⅰ.

TABLE I. RESNET50 ARCHITECTURE

convolutional layer	***output layer***	***ResNet50***
Conv-1	112×112	7×7，64，S=2 3×3 maxpool，S=2
Conv2-x	56×56	$\begin{bmatrix}1\times1 & 64\\ 3\times3 & 64\\ 1\times1 & 256\end{bmatrix}$*3
Conv3-x	28×28	$\begin{bmatrix}1\times1 & 128\\ 3\times3 & 128\\ 1\times1 & 512\end{bmatrix}$*4
Conv4-x	14×14	$\begin{bmatrix}1\times1 & 256\\ 3\times3 & 256\\ 1\times1 & 1024\end{bmatrix}$*6
Conv5-x	7×7	$\begin{bmatrix}1\times1 & 512\\ 3\times3 & 512\\ 1\times1 & 2048\end{bmatrix}$*3
1×1	Average_pool,1000-dfc,Soft_max	
Flops		3.8×10^9

III. Improved ResNet50 Image Classification

This experiment uses the PyTorch deep learning framework, researching and optimizing the classification performance of ResNet50 on the CIFAR-10 dataset based on transfer learning. The research includes four modules: first, by introducing transfer learning, pre-trained models are used to accelerate convergence; second, data enhancement techniques are applied to advance the model's capacity to generalize to unseen data; third, the loss function is optimized, and the label smoothing method is used to reduce the model's sensitivity to noisy labels; finally, the cosine annealing learning rate decay is used to accelerate model convergence and improve classification accuracy. Through these optimizations, the model's performance on small-sample datasets has been significantly improved.

A. Introduction of Transfer Learning

Transfer learning improves model learning efficiency and generalization ability by employing large-scale previously trained models with subsequent task-specific refinement [11]. The network improvement part of this paper is based on this idea, using the original network structure and parameters of the ResNet50 model pre-trained on ImageNet and applying it to new image classification tasks. Transfer learning is used to save training time as much as possible, significantly lowering the quantity of training parameters, and fine-tuning the parameters for the new model.

Assuming our original field model parameters are θs ,our target field model parameters are θt , and the fine-tuning objective can be defined as equation (1):

$$\theta_t = \arg\min \theta L_{\text{target}}\left(f\left(x;\theta\right), y\right) \quad (1)$$

The last fully-interconnected layer of this ResNet50 model is designed for the 1000-class classification task of the ImageNet dataset, with an output dimension of 1000. However, the CIFAR-10 dataset has only 10 categories. Therefore, this experiment replaces the last fully-interconnected layer of the ResNet50 model with a linear layer with an output dimension of 10, it can be expressed as equation (2):

$$model.fc = nn.Linear\left(2048,10\right) \quad (2)$$

where 2048 represents the number of input features of the fully-connected layer of ResNet50, and 10 represents the quantity of classes in the CIFAR-10 dataset. This modification ensures that the number of output categories of the model is consistent with the target task, and fully utilizes the feature extraction ability of the pre-trained model through transfer learning.

Through transfer learning, this experiment effectively reduces the time and resource costs required to train the model from scratch on the small dataset CIFAR-10, while improving the model's classification performance. This method fully utilizes the general features learned on the large-scale dataset ImageNet, enabling the model to quickly adjust to new assignments.

B. Introduction of Data Enhancement Techniques

The CIFAR-10 dataset has a relatively small number of images, containing only 60,000 images with a resolution of 32×32. If the original data is directly used to train deep neural networks, the model may not fully learn robust features, easily overfitting on the training set, leading to poor performance on the test set. Data enhancement can effectively increase the diversity of the dataset, making the model more generalizable [12-14].

Data enhancement can be mathematically viewed as perturbing the possibility distribution P(x) of the input data, simulating the changes of training data in real scenes. The enhanced data $\tilde{x}$ satisfies the following formula (3):

$$\tilde{x} = T\left(x\right), \mathrm{T} \sim \tau \quad (3)$$

Where T symbolizes a transformation function in the enhancement operation set τ ,and by applying different transformations T to the input x, diverse data samples can be generated.

For the characteristics of the CIFAR-10 dataset (resolution of 32×32, containing 10 categories), this experiment designs a series of enhancement operations to expand the diversity of training data.

1) Random Horizontal Flip

For natural images, left-right symmetrical structures are widely present. Therefore, flipping the image horizontally with a certain probability (set to 50% in this experiment) can enhance the model's adaptability to viewpoint changes.

Let the original image matrix be $X \in R^{H\times W\times C}$, where H and W are the height and width of the image, and C is the quantity of channels. The horizontal flip operation can be expressed as equation (4):

$$X^{'}_{i,j,c} = X_{i,W-j-1,c} \forall i \in [0, H-1], \mathrm{j} \in [0, W-1], \\ \mathrm{c} \in [0, C-1] \quad (4)$$

Where X' is the flipped image. This operation rotates the image symmetrically along the vertical center axis, that is, the pixel values of column j in the picture are replaced by the pixel values of columns w-j-1.

Random horizontal flipping is able to generate samples that are symmetrical to the original image but have different viewing angles, thus enhancing the variability within the training dataset and avoiding model overfitting. By introducing the flipped image, the model is able to learn the feature representation under different viewing angles, thus improving the adaptability to changes in the viewing angle of the image. In natural scenes, the left-right symmetry of objects is common. Random horizontal flipping is able to better simulate the image distribution in the real world, making the model more stable in practical applications.

2) Random Crop

In image categorization assignments, the position of objects may vary in different images, so random cropping can improve the model's robustness to target position changes. Specifically, this experiment pads 4 pixels around the image and randomly crops it to the original size (32×32), simulating the offset of the target object's position.

Let the original image size be $H \times W$,and the size of the padded image be $(H+2p)\times(W+2P)$, where p is the number of padding pixels. he random cropping operation randomly selects the top-left corner coordinates (i, j) from the padded image and crops out the target area, it can be expressed as equation (5):

$$X' = X[i : i+H, \mathrm{j} : \mathrm{j}+\mathrm{W}, :] \quad (5)$$

Where i,j follow a uniform distribution $i \sim U(0, 2P)$, $j \sim U(0, 2P)$.

3) Normalization

Normalization can adjust the numerical distribution of data, reduce the scale differences between input features, accelerate model convergence, and improve stability. Usually, we normalize each channel separately to have a value of 0 and a criterion deviation of 1, ensuring consistent data distribution across different channels.

For pixel x, the normalization process can be described as Equation (6):

$$\tilde{x} = \frac{x-u}{\sigma} \quad (6)$$

Where μ and σ represent the value and criterion deviation of the data, separately. The normalized parameters used in this experiment can be expressed as equation (7).

$$\mu = (0.5, 0.5, 0.5), \quad \sigma = (0.5, 0.5, 0.5) \quad (7)$$

This setting scales the pixel values to the interval [−1,1], improving training stability.

Through data enhancement techniques such as horizontal flipping, random cropping, and normalization, this experiment markedly advances the model's generalization ability on the CIFAR-10 dataset. These enhancement strategies effectively expand the training data distribution and alleviate the model's overfitting problem, providing important support for deep learning model training on small-scale datasets.

C. Improvement of Loss Function

In image classification, the cost function is employed to measure the goodness of the model's predictions, indicating the difference between predicted figure and the true figure. In the image classification problem of this paper, the multi-class cross-entropy cost function is generally used to measure the closeness between the real output mean and expected output mean. The closer the

actual output value is to the expected output value, the smaller the cross-entropy, and the higher the prediction accuracy [15-16]. The specific formulas of the multi-class cross entropy loss function are shown in equations (8), (9) and (10):

$$y_i = \begin{cases} 0, i \neq c \\ 1, i = c \end{cases} \tag{8}$$

$$Loss = -\sum_{i=0}^{c-1} y_i \log(p_i) = -\log(p_c) \tag{9}$$

$$Z_i^* = \begin{cases} +\infty, i = c \\ 0, i \neq c \end{cases} \tag{10}$$

In the formula: p = [p_0 , p_1 , … , p_{c-1}] represents a probability distribution, where pi is the possibility of the i-th category of samples; y=[y_0 , y_1 , … , y_{c-1}] is the one - hot representation of the sample labels; and Z_i^* represents the optimal prediction probability distribution.

When the model processes one-hot labels, it is prone to overfitting, which cannot ensure the model's generalization capacity and leads to inaccurate predictions. So, this paper employs the label smoothing approach to modify the cross - entropy loss function. Specifically, it can be defined as Equations (11), (12), and (13):

$$y_i = \begin{cases} 1-\varepsilon, i = c \\ \varepsilon \div (c-1), i \neq c \end{cases} \tag{11}$$

$$Loss_i = \begin{cases} (1-\varepsilon) \times Loss, i = c \\ \varepsilon \times Loss, i \neq c \end{cases} \tag{12}$$

$$Z_i^* = \begin{cases} log\left((c-1)(1-\varepsilon) \div \varepsilon\right) + \alpha, i = c \\ \alpha, i \neq c \end{cases} \tag{13}$$

In the equations: ε is a relatively small constant; α is an arbitrary real number; and c is the total quantity of classification types.

D. Improvement of Model Training

During the process of model training, the choice of a suitable learning rate is crucial for classification performance. This is because the learning rate decides whether the neural network has the ability to converge towards the optimal value. The commonly used learning rate decay methods for training ResNet models include equally spaced adjusted learning rate and Exponential Decay Adjusted Learning Rate.

1) Equal interval adjustment learning rate method

The model reduces the learning rate by a certain percentage after training for a certain number of iterations. This decay strategy is prone to oscillation when changing the learning rate due to the large amplitude of learning rate decay, making it difficult to converge to the optimal value.

2) Exponential Decay Adjusted Learning Rate Method

The model dynamically changes the learning rate based on the current quantity of iterations, and the quantity of iterations determines the update frequency of the learning rate. When the decay rate is small, as the quantity of iterations gradually advances, the learning rate changes less, which can easily cause the model to converge too quickly.

This paper uses the cosine annealing decay method, which reactively changes the step size in learning over time during model training. Using the characteristics of the cosine function itself to decay to the optimal learning rate, it approaches the global minimum of the Loss value, thereby allowing the model to converge to the optimal value.

The specific renewal mechanism of cosine annealing attenuation can be defined as shown in Equation (14):

$$\eta_t = \eta_{min}^i + \frac{1}{2}(\eta_{max}^i - \eta_{min}^i)\left[1 + \cos\left(\frac{T_{cur}}{T_i}\pi\right)\right] \tag{14}$$

In the formula: i is the index value; η_{min}^i is the minimum learning rate; η_{max}^i is the maximum learning rate; T_{cur} is the current quantity of iterations; T_i is total number of iterations.

IV. Experimental Results and Analysis

A. Dataset and Experimental Configuration

The experimental dataset in this paper is CIFAR-10, the OS is Win11, the processor is Intel Core i7-10750H, and the video memory is 16GB. The deep-learned framework used is pytorch 2.0, GPU device is NVIDIA Tesla T4. This experiment uses transfer learning, with ResNet50 as the base network, and adjusts its structure to adapt to the CIFAR-10 dataset. The error function is the improved cross-entropy error function, where the training set is composed of 50,000 images, while the test set consists of 10,000 images, the batch size is 64, and the training is conducted for a total of 50 epochs.

In an attempt to better objectively evaluate classification efficiency of the advanced ResNet50 network on the CIFAR-10 dataset, this paper adopts the following common evaluation metrics: classification accuracy (Accuracy), loss value (Loss), and confusion matrix (Confusion Matrix). These metrics measure the model's predictive performance from different perspectives.

Among them, classification accuracy is one of the most common evaluation metrics, which measures the ratio of accurately predicted samples to the total number of samples. For multi-classification tasks, accuracy calculation formula is shown in Equation (15):

$$Accuracy = \frac{TP + TN}{TP + TN + FT + FN} \quad (15)$$

where TP represents the quantity of precisely predicted optimistic class samples, TN represents the quantity of precisely anticipated pessimistic class samples, FP represents the quantity of samples wrongly anticipated as optimistic class, and FN represents the quantity of samples incorrectly anticipated as pessimistic class. In this paper, the accuracy of each category is calculated, and the mean value is regarded as the last evaluation metric.

Loss value is an indicator used to guide model optimization during the training process, reflecting the distinction between the model-predicted results and real labels. This paper uses the label smoothing cross-entropy loss function, the calculation approach is presented in Equation (16):

$$Loss = -\sum_{i=1}^{C} q_i log p_i \quad (16)$$

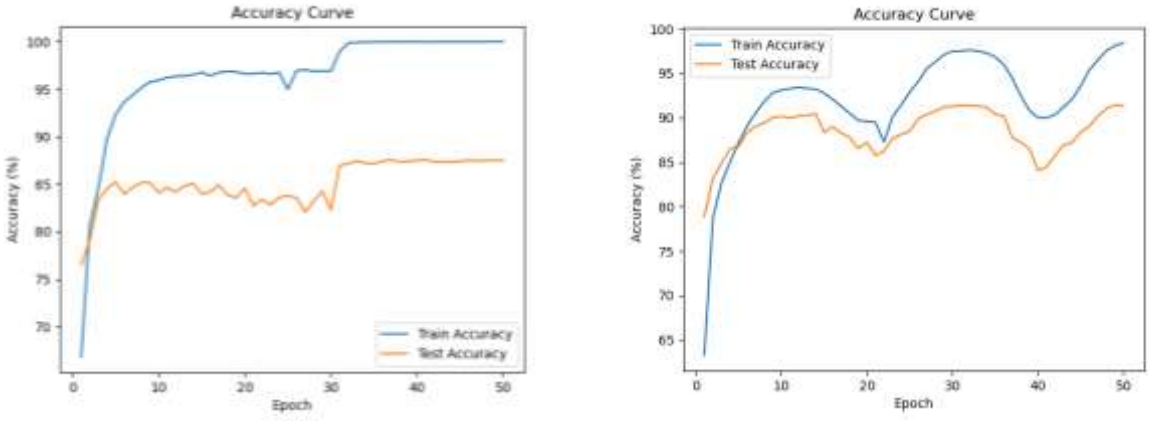
B. Assessment Criteria

Where C is the quantity of classes， q_i is target distribution after label smoothing, and p_i is the probability distribution predicted by the model. Label smoothing can reduce the model's overconfidence in certain categories, thereby advancing the model's generalization capacity. The fewer the lost-value, the greater the model's performance.

The confusion matrix is a visualization tool that intuitively displays the model's predictive performance in each category. It presents the relationship between the true labels and the predicted labels in matrix form. Diagonal values within the confusion matrix signify the quantity of correctly - labeled samples. In contrast, non - diagonal values represent the number of samples wrongly classified. In multi-classification tasks, the confusion matrix helps identify which categories are easily confused and provides guidance for model improvement.

C. Experimental Results and Analysis

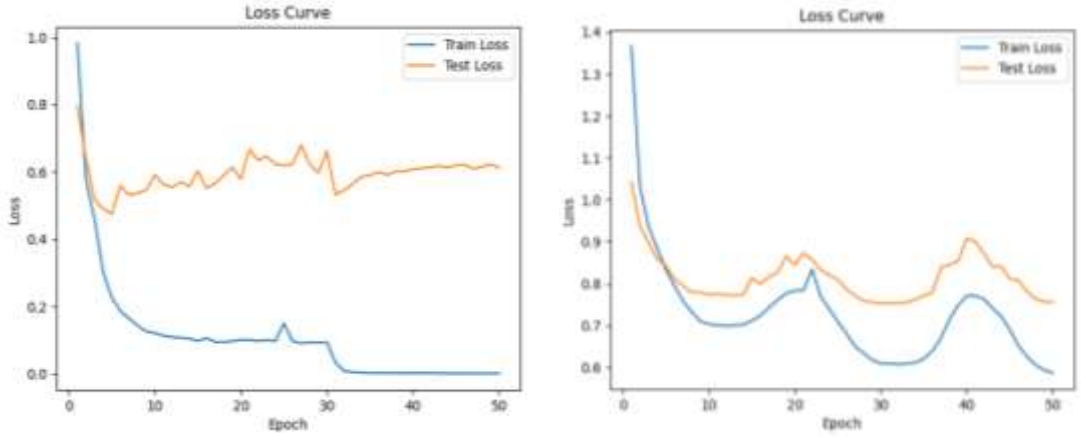
This method uses transfer learning to apply the pre-trained basic network structure, weights, and bias parameters of ResNet50 to the New-ResNet50 model, efficiently saving training time and advancing the model's generalization ability. New model is trained and tested on the information-rich CIFAR-10 dataset with 10 categories of partial image data, and finally, the model is verified to achieve better image classification ability, with a classification accuracy of 93.75% on the test set. The precision and lost-value of the original ResNet50 network and the New-ResNet50 network during testing are compared, as shown in Figure 2 and Figure 3.



（a）ResNet50 network accuracy （b）New-ResNet50 network accuracy

Figure 2. Comparison of accuracy between the two networks during training

Figure 2 illustrates the trend of validation accuracy during the coaching process for the initial ResNet50 and the improved New-ResNet50, with the horizontal axis representing the training epochs and the vertical axis representing the classification accuracy. Overall, the accuracy curve of New-ResNet50 consistently remains above that of the original ResNet50, indicating that the improved model exhibits stronger learning capability and generalization performance during training. Specifically, New-ResNet50 demonstrates significantly faster convergence in the early stages of training compared to the original ResNet50, achieving a test accuracy of 93.75% at 50 epochs, which is a 6.25% improvement over the original ResNet50's 87.50%. This validates the effectiveness of the transfer learning strategy, improved loss function, and cosine annealing decay method. However, New-ResNet50 exhibits some fluctuations between the 20th and 40th epochs, which may be related to the learning rate adjustment strategy or increased model complexity. Despite these fluctuations, the overall performance of New-ResNet50 is significantly better than that of the original ResNet50, demonstrating the effectiveness of the proposed improvements. Future work could focus on optimizing the learning rate scheduling strategy or incorporating regularization techniques (such as Dropout or weight constraints) to further enhance model stability.

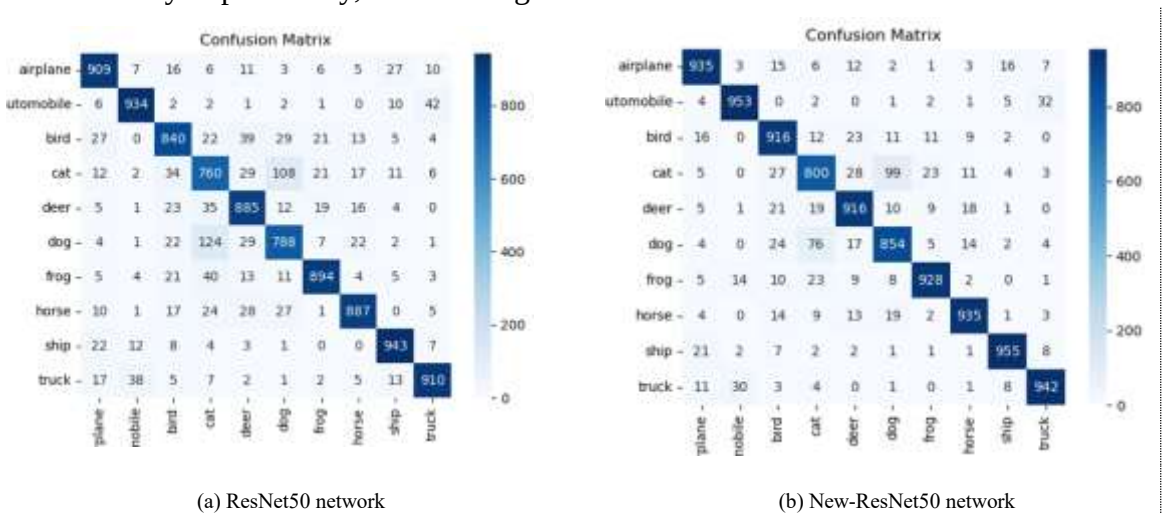


(a) ResNet50 network loss value (b) New-ResNet50 network loss value

Figure 3. Comparison of loss values between the two networks during training

Figure 3 shows the trend of loss values during the training process of the original ResNet50 and the improved New-ResNet50. The x-axis represents the number of training epochs, while the y-axis symbolizes lost-value. Overall, the training loss of ResNet50 converges faster and reaches a lower final value, indicating better fitting ability on the training set. However, overfitting is relatively evident, as the validation loss increases slightly in the later stages. In contrast, the training loss of New-ResNet50 fluctuates more significantly, showing less stable training. Nevertheless, its validation loss remains at a lower level throughout the process, with a relatively lower degree of overfitting. This suggests that the improved model has enhanced generalization ability. Specifically, the training loss of ResNet50 drops rapidly in the first 20 epochs and stabilizes after 50 epochs, with a low final training loss value. However, the validation loss increases slightly in the later stages, indicating potential overfitting. In contrast, the training loss of New-ResNet50 fluctuates more, especially between the 20th and 40th epochs. Yet, its validation loss is consistently lower than that of ResNet50 and reaches a lower level by the 50th epoch, confirming the effectiveness of the improvements in reducing overfitting.

To visually prove that the improved New-ResNet50 model has better image classification accuracy, this experiment also obtained the confusion matrix results of the two models during training, as shown in Figure 4.



(a) ResNet50 network (b) New-ResNet50 network

Figure 4. Comparison of confusion matrices between the two networks during training

Among them, the diagonal elements represent the number of correctly classified samples in each category, and the non-diagonal elements represent the misclassification situation. Overall, the New-ResNet50 network performs better on the confusion matrix than the ResNet50 network, with an increase in the number of correctly classified samples in multiple categories and a reduction in misclassification, indicating that the improvement measures have enhanced the model's classification accuracy to some degree. However, both models still have misclassification problems in some similar categories, and future work can further optimize the model, such as by increasing data diversity and improving feature extraction methods, to enhance the capacity of the model to distinguish similar categories.

V. Conclusions

This paper addresses the issues of low computational efficiency, large parameter volume, and the difficulty in balancing performance and efficiency in traditional ResNet50 models for image classification tasks

by proposing a ResNet50 image classification algorithm based on transfer learning. By employing transfer learning strategies for model initialization, introducing data augmentation techniques, and improving the loss function along with using cosine annealing decay for model training, the categorization accuracy of the model has been significantly enhanced. Experimental findings demonstrate that the suggested method improves the classification accuracy by 6.25% compared to the traditional ResNet50 image classification algorithm, achieving a classification precision of 93.75%, thereby validating the effectiveness of the proposed approach. This research not only demonstrates the significant role of transfer learning in image classification tasks but also provides new insights for optimizing deep learning models. By refining the loss function and incorporating cosine annealing decay, the study further explores optimization strategies during model training, offering theoretical references for related fields. In practical applications, the proposed method reaches a great harmony ranging from computational efficiency to classification accuracy, exhibiting extensive applicability in areas like medical image analysis, autonomous driving, and security surveillance, where it can significantly enhance system real-time performance and accuracy. Future research could further explore model lightweighting, cross-domain transfer, self-supervised learning, and multitask learning to improve model efficiency, generalization capability, and applicability, thereby promoting the application and development of deep learning technologies in more scenarios. It can be concluded that optimizing foundational models is highly necessary.

REFERENCES

[1] Gu Ruifan, Li Xiang, Ren Weimin. Research on Image Classification Based on Improved ResNet50 Model [J]. Modern Electronics Technique, 2023, 46(04): 107-112.

[2] Liu Hongda, Sun Xuhui, Li Yibin, Han Lin & Zhang Yu. A Review of Deep Learning Models for Image Classification Based on Convolutional Neural Networks. Computer Engineering and Applications, 1-29.

[3] Wang Xiuju, Fu Zhumu, Zhai Kunning, et al. Road Surface Condition Recognition Algorithm Based on Improved ResNet [J]. Science Technology and Engineering, 2024, 24(32): 14033-14040.

[4] Liu Yanru, Wu Xiaohong, He Xiaohai, et al. Research on Core Image Classification Based on Improved ResNet50 [J]. Intelligent Computer and Applications, 2025, 15(02): 10-16.

[5] Wu Di, Xiao Yan, Shen Xuejun, et al. Fruit Image Classification Based on Improved Res2Net and Transfer Learning [J]. Journal of University of Electronic Science and Technology of China, 2025, 54(01): 62-71.

[6] Jiang Zhengfeng, He Tao, Shi Yanling, et al. Remote Sensing Image Classification Based on Convolutional Attention Mechanism and Deep Residual Network [J]. Laser Journal, 2022, 43(4): 76-81.

[7] Zhang Yizhuo, Xu Miaomiao, Wang Xiaohu, et al. Hyperspectral Ground Object Classification Based on Hierarchical Fusion of Residual Networks [J]. Spectroscopy and Spectral Analysis, 2019, 39(11): 3501-3507.

[8] Fang Liang, Zhou Yun, Tang Zhiquan. Rusted Steel Bar Image Classification Based on Optimized Residual Network [J]. Journal of Northeastern University (Natural Science), 2021, 42(11): 1625-1633.

[9] Pan Renyong, Zhang Xin, Chen Xiaoyulong, et al. Apple Leaf Disease Recognition Method Based on DTS-ResNet [J]. Foreign Electronic Measurement Technology, 2022, 41(09): 142-148.

[10] Mahjoubi A M, Lamrani D, Saleh S, et al. Optimizing ResNet50 performance using stochastic gradient descent on MRI images for Alzheimer's disease classification [J]. Intelligence-Based Medicine,2025,11100219-100219.

[11] Sun H, Zhou W, Yang J, et al. An Improved Medical Image Classification Algorithm Based on Adam Optimizer [J]. Mathematics, 2024,12(16):2509-2509.

[12] Martins M F, Gonz ́alez M V, Villar R J, et al. Inception networks, data augmentation and transfer learning in EEG-based photosensitivity diagnosis [J]. Machine Learning: Science and Technology,2025,6(1):015034-015034.

[13] Chatterjee S, Byun C Y. Leveraging generative adversarial networks for data augmentation to improve fault detection in wind turbines with imbalanced data [J]. Results in Engineering,2025,25103991-103991.

[14] Yang Suorong, Yang Hongchao, Shen Furao, et al. A Review of Image Data Augmentation for Deep Learning[J/OL]. Journal of Software, 1-23 [2025-01-17].

[15] Chu Yuezhong, Wang Jiaqing, Zhang Xuefeng, et al. Image Classification Algorithm Based on Improved Deep Residual Network [J]. Journal of University of Electronic Science and Technology of China, 2021, 50(02): 243-248.

[16] Hu Kun, Wu Guoqing, Hu Zuhui, et al. Research on Metal Surface Defect Image Classification Based on Improved VGG16 Network [J]. Computer Applications and Software, 2024, 41(06): 175-180.

Research on Crop Detection Algorithm Based on Improved YOLOv7

Xiaoqi Shi
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: shixiaoqi713@163.com

Xin Ye
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: yexin@xatu.edu.cn

***Abstract*—In the field of crop target detection, traditional target detection algorithms are often difficult to achieve satisfactory accuracy due to factors such as dense distribution of species and poor imaging quality, which brings many inconveniences and challenges in practical agricultural production applications. To address this situation, the study introduces an enhanced YOLOv7 algorithm, incorporating the attention mechanism, with the objective of substantially elevating the overall performance in crop target detection tasks. The improved algorithm can more accurately focus on the key features of crops by cleverly incorporating the attention mechanism, effectively filtering out the interference of complex background and noise, so as to achieve more accurate recognition of various crops. After a large amount of experimental data verification, the improved algorithm can achieve an average recognition accuracy of 80% for a variety of crops, with an average accuracy of 75%, and the highest recognition efficiency is as high as 91% in the detection of some specific crops. In contrast to other prominent crop target detection algorithms, the refined algorithm presented in this paper exhibits remarkable performance benefits. Notably, its target detection efficacy is highly significant, enabling swift and precise identification of crop species.**

Keywords-Target Detection; Attention Mechanism; YOLOv7; Crop Species Recognition

I. INTRODUCTION

Our country is a country dominated by agriculture in historical records, and fruits and vegetables have an indispensable position in the everyday lives of our populace. In recent years, China's fruit and vegetable industry has developed rapidly, with the development of domestic vegetable farming and the continuous introduction of foreign fruits and vegetables. Crops in quantity, quality and category also meet the growing demand of urban and rural residents, which makes China's fruit and vegetable production and sales are also increasing year by year. However, in recent years, more and more people have been farther and farther away from agricultural production, resulting in a lack of knowledge about common crops. The conventional approach to identifying crop species primarily involves manual visual inspection, which is not only inefficient but also constrained in accuracy by the inspectors' experience and skill levels.

In recent years, China's fruit and vegetable industry has undergone swift development, accompanied by a steady influx of exotic fruits and vegetables， in terms of quantity, quality and variety of categories to meet the growing needs of urban and rural residents, a large number of fruit and vegetable products break through the original cognition of the people, and therefore urgently need related technology to help people quickly identify the relevant products [1]. Due to the swift advancements in machine vision and artificial intelligence, AI technology has progressively integrated into various facets of production and daily life. Intelligent classification and identification of fruits and vegetables by means of deep learning machine vision is an ideal way of processing [2].

In this study, the model's foundational architecture is chosen to be the YOLOv7 algorithm, and to augment its recognition precision, the SE-Net attention mechanism is integrated into its main network structure. Following these enhancements, the YOLOv7 model's performance in crop recognition tasks has

undergone notable improvement in accuracy, and all the experimental results meet the expected goals. These improvements effectively validate the initial idea of performance enhancement of the model.

II. Related Work

Foreign research in fruit and vegetable identification began at an earlier period, foreign scholars used LiDAR technology and deep learning to predict vegetable crop growth, combined with LiDAR (laser radar) technology, formulated a deep learning-based prediction model, capable of estimating the height and canopy size of vegetable crops. On an experimental farm at Bangalore Agricultural University in India, data pertaining to the growth cycles of tomatoes, eggplants, and kale were gathered across five distinct time points spanning a specific period. The research team at Bangalore Agricultural University, India, used a terrestrial laser scanner to acquire LiDAR point clouds and integrated a hybrid deep learning architecture that merges Long Short-Term Memory Networks (LSTMs) with Gated Recurrent Units (GRUs).to make predictions [3]. The deep learning model exhibited an approximately 80% accuracy in anticipating structural parameters at the plant level for various stages of crop growth in advance. Specifically, the hybrid model demonstrates efficacy in forecasting canopy area, with height prediction errors ranging from 5% to 12%, and a balanced occurrence of both over- and underestimation biases.

Domestic research in the direction of fruit and vegetable identification started relatively late, and most of them only studied and improved the algorithm. In 2016, Zeng Weiliang et al. designed a fruit and vegetable recognition system for smart refrigerators through convolutional neural networks, which is based on the improved LeNet-5 algorithm, on which the ReLU activation function is used and the Droupt technique is used. It was tested on a dataset with 15 categories of fruits and vegetables and a total of 2633 images of fruits and vegetables, and the experimental results obtained an accuracy of 83.4% [4]. In addition, in 2020, by Cheng Shuai, Li Yanling, Si Haiping, and Sun Changxia, the network parameters were optimally adjusted based on the DenseNet121 network. It was trained and tested on a kind of dataset with only five types of crops and compared with the classical VGG16 model and ResNet50 model on the same dataset, and the evaluation results indicated an enhancement in the network's recognition rate by 1.1% and 6.9%, when compared to the two respective algorithms [5].

Other scholars have used deep learning algorithms for feature extraction, spot segmentation, and detection and recognition of different disease classes on crop leaves. To identify crop leaf diseases, they employed diverse deep learning techniques, including convolutional neural networks (CNNs) and support vector machines (SVMs), among others. Through the construction of a deep learning model, followed by its training and optimization processes, high accuracy recognition of crop diseases was achieved [6]. The deep learning-based crop disease recognition method is faster and simpler than the traditional recognition method, and the recognition accuracy is improved. Some studies have shown that deep learning-based plant disease classification and recognition methods can achieve 91% to 98% recognition accuracy.

III. Technical model

A. YOLOv7 Algorithm

YOLOv7 is a network model in the YOLO family introduced in recent years. This model is currently the YOLO model with the fastest inference speed and best recognition results on the PASCAL VOC dataset. When compared to other target detection models, it surpasses in both speed and accuracy, thereby fulfilling the requirement for prompt and precise identification of crop species in natural settings. However, although the original YOLOv7 algorithm has already attained high detection accuracy, there remains potential for enhancement to mitigate the influence of similar features, target scale and shape variations on the detection accuracy. YOLOv7 proposes a new network architecture called ELAN (Efficient Layer-wise Adaptive Network), which focuses on high efficiency. The ELAN framework facilitates efficient learning and convergence in deeper networks by regulating the gradient paths, whether they are the shortest or longest [7].

YOLOv7's loss function comprises components for coordinate error, confidence in target prediction, and classification accuracy. The matching strategy employs the SIMOTA method, which obtains anchor frames by k-means clustering and positive sample expansion. The network architecture of YOLOv7 consists of the CBS (Conv, BatchNorm, Silu) module, the CBM module, the REP module, the MP module, the ELAN module, the ELAN-W module, the UP Sample module and SPPCSPC modules. These modules work together to achieve efficient feature extraction and target detection [8].

YOLOv7 uses Leaky ReLU (Rectified Linear Unit with Leakage) as an activation function [9], which solves the problem of the traditional ReLU function having zero gradient in the negative region.

It is due to the above characteristics that the YOLOv7 algorithm has been widely used in application scenarios where the demand is for high speed and high accuracy, and it also meets the criteria for speed and precision in detection in this crop species identification system. Therefore, this study utilizes YOLOv7 as the foundational detection model and further refines it. The YOLOv7 network architecture is shown in Figure 1.

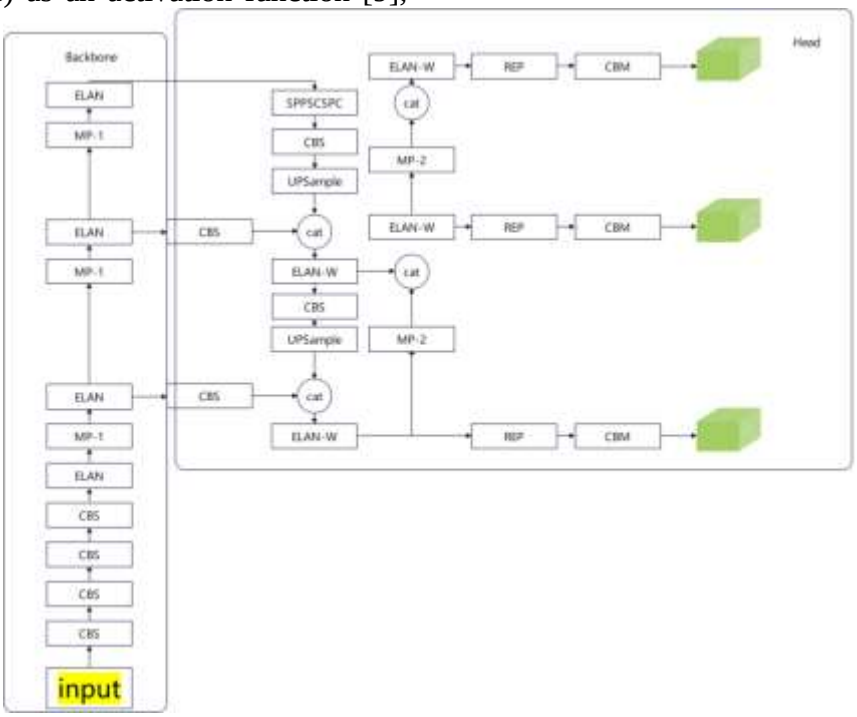


Figure 1. YOLOv7 network architecture diagram

In order to further improve the detection accuracy of the model, this study introduces an enhanced algorithm built upon the YOLOv7 framework, specifically, the method of adding an attention mechanism to the Backbone network to enhance the precision of detection using the YOLOv7 algorithm [10], by comparing the model detection accuracy before and after the improvement.

The Backbone network in the YOLOv7 algorithm is mainly composed of two major modules, Multi_Concat_Block module and Transition Block module [11].

1) Multi_Concat_Block Module

There are four feature layers that perform the final feature stacking in this module, the 0th bit labeling, the 1st bit labeling, the 3rd bit labeling, and the 5th bit labeling. Bit 0 does not operate, bit 1 performs 1 convolution, bit 3 performs 3 convolutions, bit 5 performs 5 convolutions, and after feature stacking the features are integrated by one convolution, in this module, the main

operation performed is bitwise convolution. As shown in Figure 2.

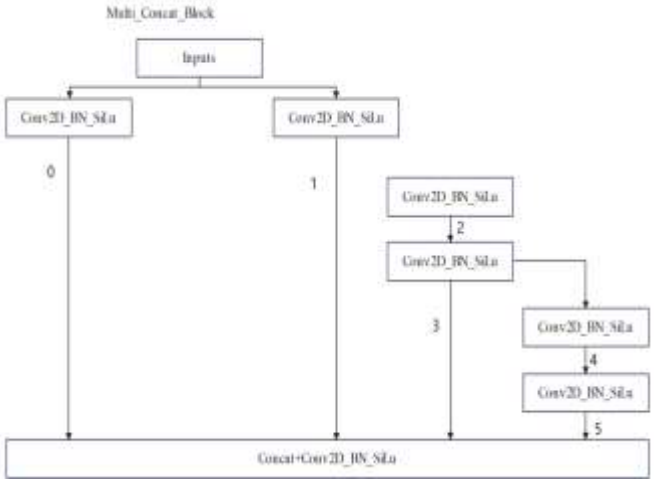


Figure 2. Schematic diagram of Multi_Concat_Block module

2) Transition Block Module

The module consists of two main branches, the left branch of the input data into a step for the maximum pooling, and then through a convolution to adjust the number of channels from 1024 to 512. the right branch of the input data into a convolution and then through a convolution kernel size of the step for the convolution of the extraction of features. Refer to Figure 3 for illustration.

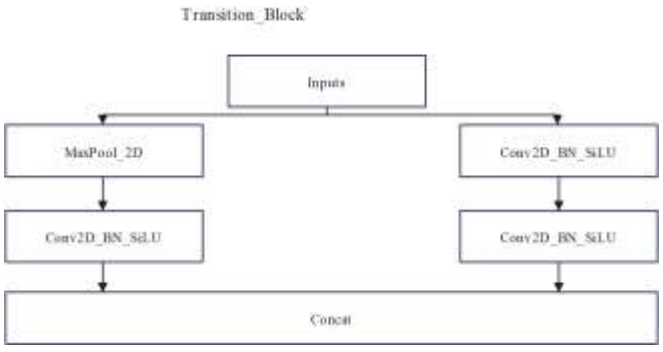


Figure 3. Schematic diagram of Transition Block module

3) FPN Enhanced Feature Extraction

The role of SPPCSPC module is to increase the sensory field, the module is mainly divided into two branches, the left branch of the first four branches were 5, 9, 13, 1 max-pooling [12], this process enables four sensory fields to distinguish between large and small objects. For example, there are different sizes of fruits and vegetables in a photo, their sizes are not the same, after this module will be able to better distinguish between small and large targets, SPPCSPC structure sketch depicted in Figure 4.

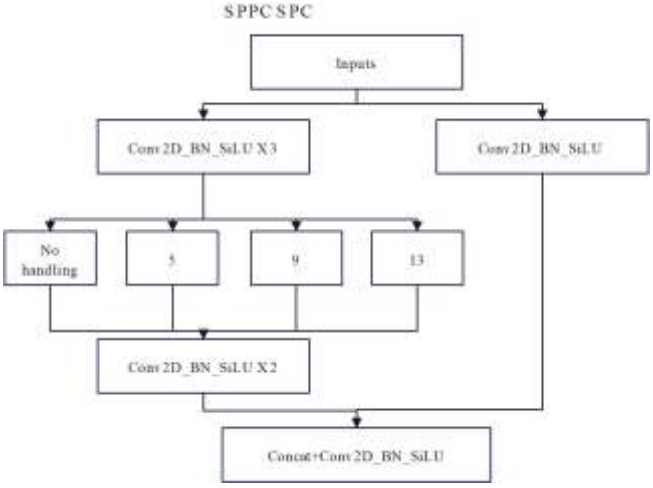


Figure 4. Sketch of SPPCSPC structure

The following are the steps for feature fusion at the FPN layer.

Step1. Using SPPCSPC for feature extraction to operate on the bottom feature layer can improve the sensory field of YOLOv7 algorithm, making YOLOv7 algorithm can be more accurate in recognizing objects with different sizes within the picture.

Step2. Conduct 1×1 convolution to adjust the channel, and then perform up-sampling operation to combine with the feature layer after one convolution of the middle and lower feature layers, and then use Multi_Concat_Block module for feature extraction.

Step3. Perform 1×1 convolution to adjust the channel for the feature layer obtained in the second step, then perform up-sampling and combine it with the feature layer of the middle layer after one convolution, and then use the Multi_Concat_Block module for feature extraction.

Step4. The feature layer obtained in the third step is down sampled by the Transition Block module once, and then down sampled and stacked with the feature layer obtained in the second step,

after which feature extraction is performed using the Multi_Concat_Block module.

Step5. The feature layer obtained in the fourth step is down sampled by the Transition Block module once, and then stacked with the feature layer obtained in the first step after down sampling, and then the Multi_Concat_Block module is used for feature extraction.

Up to this point, three enhanced feature layers are obtained, which are the feature layers obtained in the third, fourth and fifth steps, respectively. The feature pyramid can complete the feature fusion operation between feature layers of different shapes, which is helpful to extract better features [13]. The feature layer shape change diagram is shown in Figure 5.

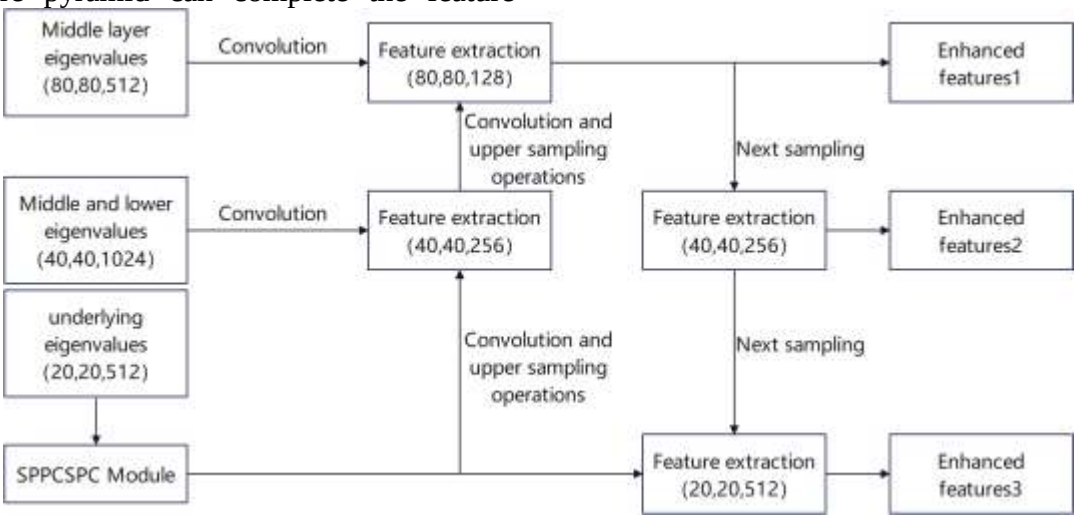


Figure 5. Feature layer shape change map

Add SE-Net attention mechanism, its full name is Squeeze-and-Excitation Networks, YOLOv7 algorithm is still a kind of algorithm based on the improvement of CNN in essence [14], and SE-Net attention mechanism is a kind of model that introduces the attention mechanism in the convolutional neural network.

The attention mechanism in SE-Net is mainly realized through SE Block, the specific procedures are outlined below.

Step1. Squeeze Compression. The SE-Net attention mechanism employs global average pooling to reduce each channel's feature maps to a single value, capturing global spatial information across channels. The dimensionality of this step is varied as $(C,H,W)\to(C,1,1)(C,H,W)\to(C,1,1)$.

Step2. Excitation. The SE-Net attention mechanism uses two fully connected (FC) layers to learn the relationships between channels, generating a weight for each channel. The first fully-connected (FC) layer performs a dimensionality reduction and the second fully-connected (FC) layer reverts to the original dimensions and an activation function is applied to guarantee positive weights summing to one. The dimensionality change for this step is $(C,1,1)\to(C,1,1)(C,1,1)\to(C,1,1)$.

Step3. Scale deflation. Multiply the weights obtained from the motivation step by the original feature map, the dimensionality change of this step is $(C,H,W)\times(C,1,1)\to(C,H,W)(C,H,W)\times(C,1,1)\to(C,H,W)$

The attention mechanism is introduced in the three intermediate layers, the lower middle layer and the bottom feature layer positions in the Backbone network mentioned earlier, and the specific introduction position of the attention mechanism is shown in Figure 6.

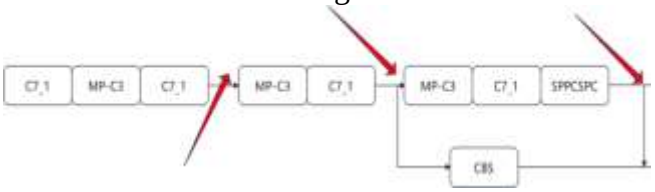


Figure 6. Map of the location of the introduction of the attention mechanism

IV. Experiment and Analysis

B. Experimental Content

Target detection of crop species, in the establishment of the data set, take the fruit and vegetable photos obtained on the Internet to build their own data set, the pictures include multiple angles, as well as different lighting conditions and increase the interference term and other forms of sampling, and then organize the experimental data set suitable for the needs of the experiment, and ultimately a total of seven common vegetables as well as seven types of fruits to form a data set, a total of 14 different types of Crops. To assess the method's performance across various training iterations in this study, experimental demonstrations were conducted on this data set.

Compared to traditional image recognition methods, the YOLO algorithm introduced in this study demonstrates superior results in detecting crop species targets The refined YOLOv7 method achieves notable enhancements in processing speed as well as detection precision.

The environment for this experiment is the server operating system is Ubuntu 20.04, the number of CPU cores is 8 cores, RAM is 15G, and the GPU is GeForce RTX 2080 Ti with 11G of video memory. To speed up the training time, the GPU is used for acceleration, and the code is written using the Python 3.9 programming language. The constructed dataset was labeled using Labellmg software. SSH connection was used to connect the local machine to the cloud server, and Xftp7 software was used to upload project code and files to the cloud server, as well as to download the trained models.

C. Experimental Process

The roadmap for the realization of the experiment is shown in Figure 7.

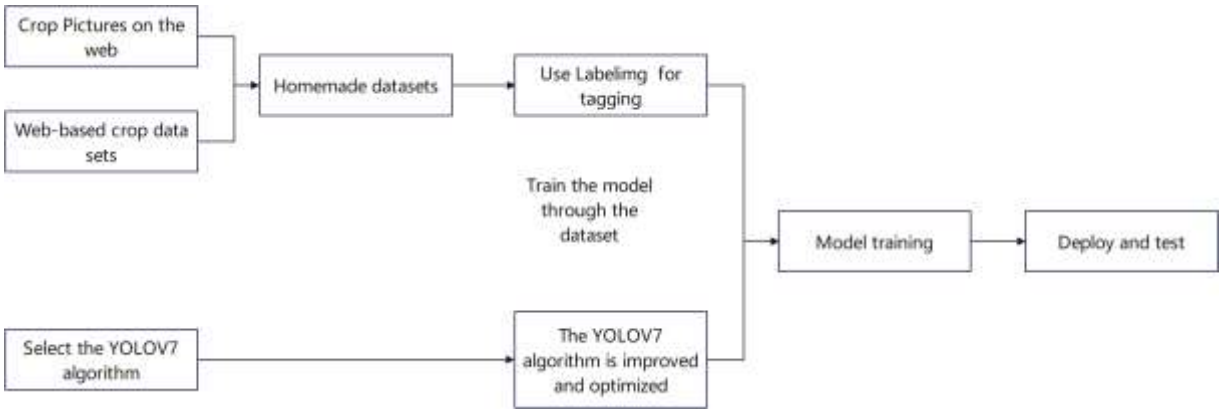


Figure 7. Roadmap for system realization

This paper uses a homemade dataset with specific types and quantities of fruits and vegetables as shown in Table 1.

TABLE I. Table of fruit types and corresponding number of pictures

Name and number of vegetables	Name and number of fruit
Cabbage（200）	Apple（200）
Capsicum（200）	Banana（200）
Carrot（200）	Pear（200）
Cauliflower（200）	Pineapple（200）
Corn（200）	Pomegranate（200）
Eggplant（200）	Grapes（200）
Cabbage（200）	Apple（200）

In the preprocessing stage, the homemade dataset was image labeled using Labellmg image labeling software, and the dataset was allocated into training, validation, and testing subsets in an 8:1:1 ratio.

D. Experimental Results

The models before and after the improvement are trained in the same dataset and experimental environment framework in the experiments, and whether the improvement has improved the accuracy is obtained by comparing the detection accuracy of the two models.

The confusion matrix obtained here is a 14×14 matrix as there are 14 categories in total.

In the matrix, rows signify actual categories while columns represent predicted categories. Diagonal elements indicate correct classifications, whereas off-diagonal elements signify misclassifications in the matrix.

The confusion matrix plot obtained at the end of training is shown in Figure 8. Where banana, cabbage, pepper, cauliflower, corn, pear, pineapple and pomegranate have a value of 1.00 on the diagonal, which indicates that the model is better trained for these eight types of crops, while the other types of crops have the problem of prediction error during model training, where the grapes have the largest prediction error of 0.50. this indicates that this model has a lower prediction for grapes.

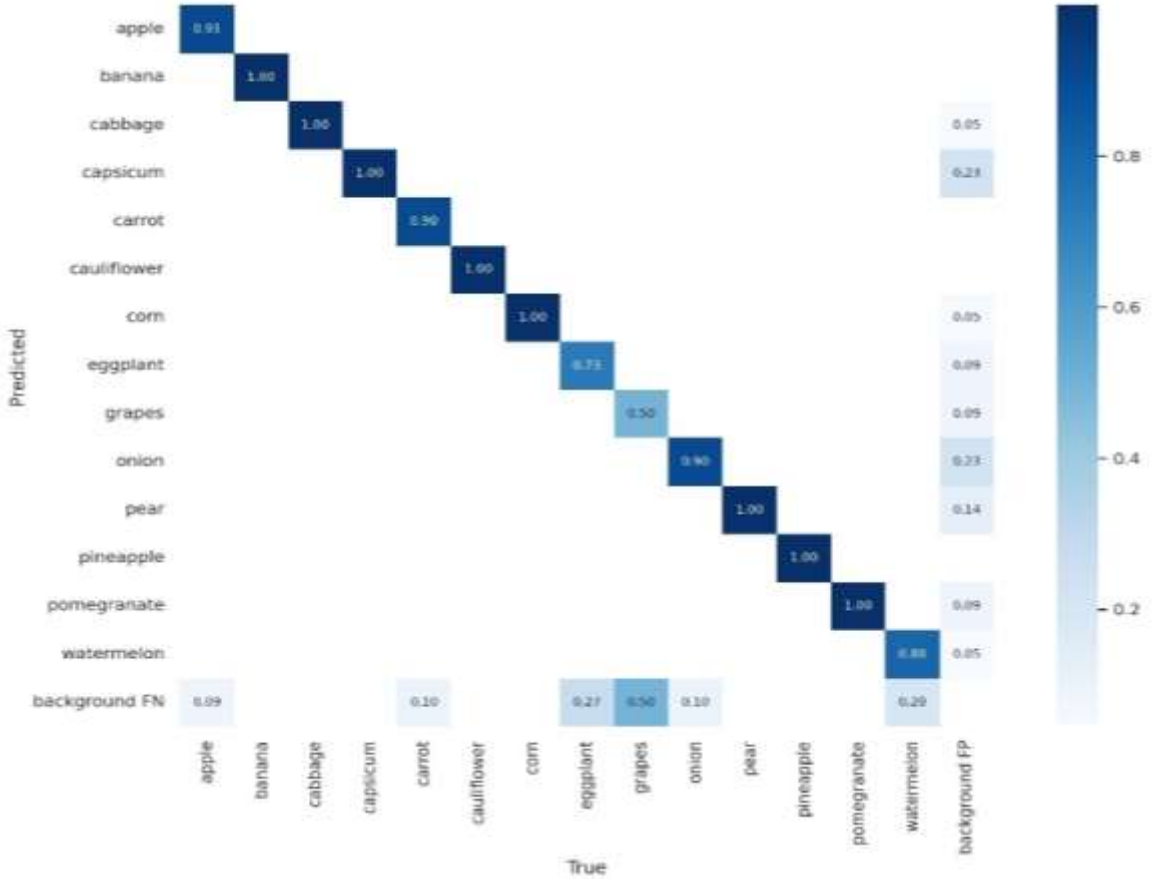


Figure 8. Confusion matrix diagram

The reconciled mean of checking accuracy and recall is defined as F1, with a maximum of 1 and a minimum of 0. Typically, a greater F1 score indicates superior model performance, as illustrated in Formula (1).

$$F1 = \frac{P \times R}{P + R} \times 2 \quad (1)$$

Figure 9 displays the F1 images, with the horizontal axis depicting various thresholds and the vertical axis showing the corresponding F1 scores, and the altered image indicates that the F1 scores of all the species were maximized in the interval from 0.80 to 0.417.

The experiment's P_curve graph illustrates the correlation between accuracy and confidence, a prevalent method for visualizing target detection outcomes. The P-curve plot is advantageous as it facilitates the assessment of detector performance and the determination of an appropriate threshold by depicting accuracy curves across varying confidence levels. Figure 10 illustrates this concept. The plot's horizontal axis depicts the confidence level, while the vertical axis represents the accuracy rate. Each thin line on the plot

corresponds to the accuracy curve of a specific category, and the thick line represents the mean precision curve across all categories. The confidence level, if too high, may miss some real samples with low determination probability. From Figure 10 below, we can see that at a confidence value of 0.942, the model in this paper achieves perfect accuracy (i.e., no false alarms) for all categories, which is in line with the trend that this figure should have, and the results are better.

The PR curve of the experiment represents the relationship between the precision and the recall. The PR curve serves to demonstrate the model's performance across varying thresholds by plotting the precision and recall rates as the decision thresholds change, and the performance of the model is better indicated when the curve is positioned closer to the upper right corner. The PR curve of the training results of the experiments in this paper is shown in Figure 11, on the PR curve, the recall rate is plotted along the horizontal axis, while the precision rate is represented on the vertical axis. AP denotes the area under the curve, mAP refers to the average of the average precision (AP) of all target categories in the model, and the threshold used to classify IoU as a positive or negative sample is denoted by the number following mAP@. mAP@0.5 denotes the mean mAP for which the threshold is greater than 0.5. The calculation is shown in Formula (2).

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \qquad (2)$$

Where N in the formula indicates the number of all categories, in this experiment N=14, which is indicated as the mean precision of the i-th category, the mean precision value can be obtained by adding the accuracy values of its 14 categories and dividing by 14. It can be seen that the mAP@0.5 of target detection of this paper's model is 0.822, and the obtained experimental results align with the anticipated outcomes.

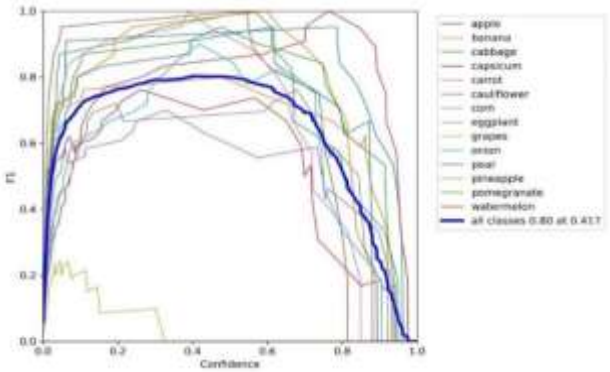


Figure 9. F1 score graph

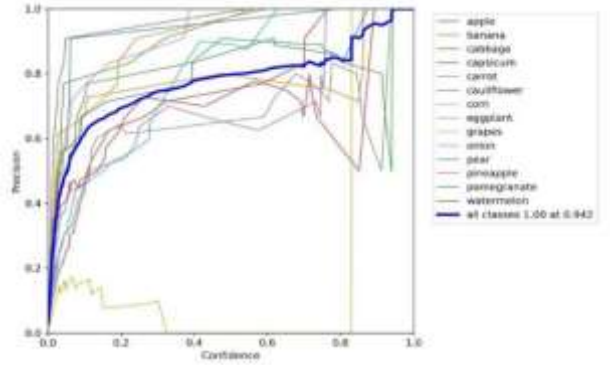


Figure 10. P_curve

The top right corner of Figure 11 below shows the 14 crop categories and their respective accuracy values during training.

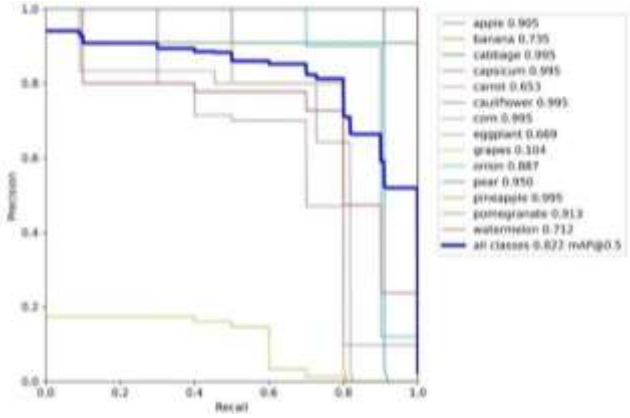


Figure 11. PR Curve

The R_curve plot of the experiment represents the relationship between recall and confidence, and the function of the R_curve plot is to understand the model's ability to recognize positive samples under different conditions by plotting the recall curves under different confidence levels. As shown in Figure 12. The

horizontal coordinate of the graph is the confidence level and the vertical coordinate is the recall, where the thin line represents the recall curve for each category and the thick line represents the average recall curve for all categories. Ideally, when the confidence threshold is 0, all the detected frames are retained, and thus the recall should be 1. However, in practice, factors such as errors in the target detection algorithms and noise may cause some detected frames to be incorrectly filtered out, which results in a recall that is not 1.

At lower confidence levels, categories are detected more inclusively. Specifically, when the confidence level is set to 0, the average recall across all categories achieves 0.95, surpassing experimental benchmarks and fulfilling requirements.

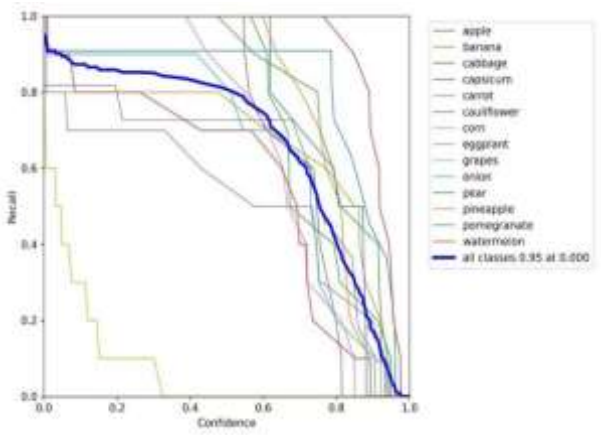


Figure 12. R_curve

Figures 13 and 14 present the experimental outcomes, depicting results for various crops.

Figure 13. Apple experiment result

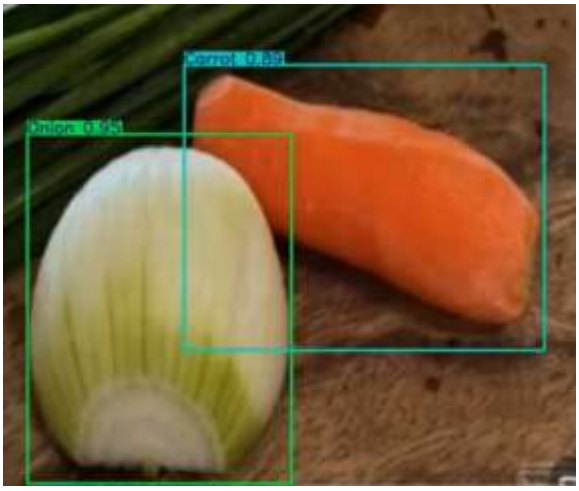


Figure 14. Results of Onion and Carrot experiments

After detecting a large number of images, the detection accuracy values obtained from each detection are recorded and the resulting detection accuracy is shown in Table 2.

TABLE II. COMPARISON TABLE OF DETECTION ACCURACY

Type	Evaluation metrics		
	Detection Times	mAP/% (Pre-improved)	mAP/% (Improved)
Apple	30	0.79	0.85
Banana	30	0.74	0.77
Pear	30	0.79	0.81
Pineapple	30	0.81	0.79
Pomegranate	30	0.68	0.68
Grapes	30	0.50	0.57
Watermelon	30	0.73	0.78
Cabbage	30	0.89	0.91
Capsicum	30	0.55	0.58
Carrot	30	0.83	0.89
Cauliflower	30	0.55	0.64
Corn	30	0.46	0.45
Eggplant	30	0.74	0.71
Onion	30	0.88	0.95

By summarizing the data in Table 2 above, it can be seen that the improved model is higher than the pre-improved model in all the other 11 categories, although three categories are lower

than the pre-improved one in terms of detection accuracy.

V. CONCLUSIONS

For each crop, the average value of the detection accuracy for each category is different, both before and after improvement, which is due to the fact that when training the model, the detection of some categories is not very satisfactory due to some external factors, such as incomplete image annotation when building the dataset on its own, and thus the detection of some categories is not very satisfactory. In this paper, a fruit and vegetable dataset suitable for this study is constructed by collecting and organizing the data, including seven kinds of vegetables commonly found in supermarkets and farmers' markets as well as seven kinds of fruits, with a total of 14 categories and 3220 images. In this experiment, the YOLOv7 algorithm serves as the foundational network for the entire model, and the SE-Net attention mechanism is added to its backbone network to improve the accuracy of the model. It makes up for the past defects such as low accuracy of crop recognition, etc. Using the improved model to train the dataset, several sets of experiments were conducted, and the results of the experiments were analyzed and discussed to conclude that the improved approach of this study has a positive effect on this experimental dataset.

The improved YOLOv7 model achieved an average accuracy of 80% and an average precision of 75% in crop species recognition. All the experimental results met the envisioned expected values. In future research, further lightweighting studies of the network model will be conducted to enhance the model's detection rate.

REFERENCES

[1] ZouJunRong. 2022 Fruit and Vegetable Industry Development Outlook and Market Trend Research and Analysis. https://www.chinairn.com/scfx/20220715/121648626.shtml, 2022-7-15).

[2] Pan Mei. Application of image recognition in fruit and vegetable classification and recognition[J]. Modern Agricultural Science and Technology,2021(16):257-259.

[3] J, R., Nidamanuri, R.R. Deep learning-based prediction of plant height and crown area of vegetable crops using LiDAR point cloud[J]. Scientific Reports,14,14903(2024).

[4] Zeng Wei-liang, Lin Zhi-xian, CHEN Yong-shan. Research on fruit and vegetable image recognition for smart refrigerator based on convolutional neural network[J]. Microcomputer and Applications,2017,36(08):56-59.

[5] Cheng Shuai, Li Yanling, SI Haiping, et al. Research on automatic crop species identification algorithm based on convolutional neural network[J]. Henan Science,2020,38(12):1908-1914.

[6] B. V S, Harshad B, Vijay D. Hunger games search based deep convolutional neural network for crop pest identification and classification with transfer learning[J]. Evolving Systems,2022,14(4):649-671.

[7] Zhao Pengfei, Qian Mengbo, Zhou Kaiqi, et al. Improvement of sweet pepper fruit detection in YOLOv7-Tiny farmland environment[J]. Computer Engineering and Application,2023,59(15):329-340.

[8] Wang Yihan. Research on citrus fruit recognition and localization method in natural environment based on improved YOLOv7[D]. Sichuan Agricultural University,2023.

[9] Wu Jie, Shi Lei, Zhang Zhi-An. Research on pest image recognition and classification method based on deep learning[J]. Computing Technology and Automation,2023,42(01):166-173.

[10] Luo Tonglan. Research on potato defect detection based on improved YOLOv7[D]. Ningxia University,2023

[11] Pang Haitong. Research on intelligent identification technology of orchard pests based on deep learning[D]. Zhejiang University,2021.

[12] Fang Si-Wen. Research on apple localization and identification technology in complex environment based on YOLOv7[D]. Henan Agricultural University,2024.

[13] Jian Huang, Gang Zhang. A review of target detection algorithms for deep convolutional neural networks[J]. Computer Engineering and Application,2020,56(17):12-23.

[14] Xu Qiu. Research on transmission line bird damage detection and hazardous bird species identification based on YOLOv7[D]. Nanchang University,2024.

Research on Automatic Problem-Solving Technology of Olympic Mathematics in Primary Schools Based on AORBCO Model

Sijie Wu
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: 2291228142@qq.com

Wuqi Gao
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: gaowuqi@xatu.edu.cn

Liping Lu
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: llp21@126.com

***Abstract*—This study addresses intelligent problem-solving in elementary math competitions by proposing an AORBCO model-based system. It integrates knowledge graphs, rule-based reasoning, and cognitive optimization to simulate human problem-solving processes. The framework systematically analyzes competition problem types, constructs a structured knowledge base, and implements dual-solving modules: rule-template matching and knowledge graph reasoning, supplemented by question bank similarity retrieval. Experimental results demonstrate 15% higher accuracy and 30% faster solving speed compared to conventional methods, with enhanced interpretability. Key innovations include the first application of AORBCO in educational AI, novel knowledge representation methods, and specialized cognitive optimization algorithms. The research provides technical support for personalized math education and advances intelligent tutoring systems. Future work will focus on improving model generalization and exploring multimodal learning integration.**

Keywords-Automatic Rule Generation; Ego Agent; Automatic Reasoning; AORBCO Model; Autonomous Learning Ability

I. RESEARCH STATUS

A. Semantic Understanding of Natural Language Topics

Considering the importance of semantic understanding in the process of problem analysis, the Ego individual needs to truly clarify the meaning of the natural language received, which refers to the knowledge described by humans using natural language, thereby updating its own cognition based on this understanding. The degree of the Ego's understanding of natural language relies on the prior knowledge possessed by the current Ego, similar to the process of human enlightenment learning. However, regardless of whether the Ego's understanding is correct, the form of the understanding's representation is expressed in the form of knowledge described using descriptive language. The Ego first processes the received natural language text sentence by sentence, based on the prior knowledge it possesses, splits the sentences, and understands the sentences word by word using the concepts of nouns. The Ego then comprehends the semantics of the entire sentence and finally uses the AORBCO model to represent the semantic information understood by the Ego [1].

B. Automatic Reasoning

The methods of reasoning are the current automatic problem-solving software for elementary geometry. Due to the limitation of data structure and reasoning methods, most of them are based on one-way application, and most of them are forward deduction. The advantage of forward deduction

DOI: 10.2478/ijanmc-2025-0013

method is that a lot of useful information can be inferred from the known information whether the conclusion can be deduced or not, which is of great significance to inspire students to think; The disadvantage is that the efficiency is not ideal for the reasoning of topics with more known information [2]. The backward inference method is suitable for the situation that there are a lot of known information and there are few goals to prove. Its main advantages are that it is not necessary to use information that has nothing to do with the goal, and it is beneficial to provide explanations to users. The disadvantage is that the selection of sub-goals is blind and affects efficiency. The advantages and disadvantages of reasoning with only one method are obvious, so consider realizing a combined system to make it have the advantages of both forward and backward reasoning systems, which is a two-way reasoning system. Two-way reasoning overcomes the shortcomings of weak purpose of forward push and blind choice of target of backward push, and at the same time combines the advantages of both. The realization technology is relatively more complicated than the single system, and some difficult problems mainly lie in the judgment of the joint point of forward and backward push, the proportion distribution of forward and backward push and so on [3].

The basic idea of realizing two-way reasoning system is: forward reasoning according to known facts but not all the way to the target (otherwise it is a forward reasoning system); At the same time, backward reasoning from the goal is not always until the known facts are reached (otherwise it is a backward reasoning system) [4]. Combining these two kinds of reasoning in some intermediate link between known facts and goals is the condition for the successful termination of two-way reasoning.

II. The Definition of Intelligent Reasoning Based on Ego

Knowledge is constantly changing in the system of human mind. Among them, knowledge has a very important changing factor, that is, knowledge will be reduced by human memory according to the increase of time, that is, we often say that knowledge will be "forgotten"

That is, if a knowledge is first remembered by human beings (or stored in the knowledge base by Ego for the first time), then if this knowledge is not "reviewed" (or recalled by Ego), this knowledge will gradually be "forgotten" by human beings (or forgotten by Ego). This law is the most basic evolutionary factor in the renewal and evolution of knowledge [5], so it is called the basic weight of knowledge in AORBCO model and expressed by Bi. The knowledge in AORBCO model has the attribute of weight, and the basic weight Bi is one of the factors that make up weight through calculation. The calculation formula of Bi is as follows:

$$B_i = 100k / \left(\left(\log t_i \right)^c + k \right) \quad (1)$$

This formula is based on Ebbinghaus's original data and the forgetfulness curve fitted by the researcher. Where k=1.84, c=1.25, and ti is the time interval between this recall and the last recall. If a knowledge is successfully recalled when Ego receives the current wish, the ti of this knowledge will be updated to the latest value of 1 according to Ebbinghaus curve [6]. That is what we commonly call "every time you study, your knowledge will be consolidated in your mind". It should be noted that, according to the previous discussion in this paper, when people use a certain knowledge, they will "associate" with other knowledge, which is also the core issue of knowledge evolution in this paper [7]. When Ego recalls a certain knowledge according to this wish, other knowledge related to this knowledge will also be "associated" by Ego and reviewed for 46 times. Therefore, the associated knowledge will also evolve according to Ebbinghaus curve according to the semantic distance coefficient from the recalled knowledge.

The AORBCO model is centered around the Ego, with the knowledge within the model being a reflection of the Ego itself. To achieve the intelligence of the AORBCO model and to align its behavioral activities more closely with the intelligent mechanisms of human cognitive thinking, research is conducted to improve the AORBCO model by studying the four characteristics of intelligent self-awareness [8], mutual representation, ambiguity, and dynamism.

Additionally, a descriptive language for the AORBCO model is designed to provide a clearer and more explicit description of the model's theoretical concepts and structural components. By analyzing human intelligent thinking activities and drawing lessons from the human problem-solving process, this research abstracts human cognitive thinking activities and reflects them in the model [9]. The improved AORBCO model characterizes the self-awareness of intelligence through five core components: beliefs, capabilities, desires, planning, and behavioral control mechanisms; it represents the mutual representation of intelligence starting from entities, including the familiar subjects of agents and the objects they recognize; it introduces weights that indicate the closeness of relationships between entities, simulating the ambiguity of intelligence through changes in relational weights; and finally, it implements the operation of the model through behavioral control mechanisms, allowing its behavioral activities to influence the other components, thereby realizing the dynamism of intelligence.

The main focus of this paper is the matching reasoning module of the problem-solving system. The reasoning engine employs traditional forward reasoning methods, integrating computation into the reasoning process to iteratively generate new knowledge. Matching reasoning primarily utilizes the resolution principle of first-order predicate logic. The rules in the system consist of first-order predicate logic, with clauses containing variables, meaning that predicates have direct relationships and are governed by the semantics of predicates. By matching the variables in the rules with the entities understood in the problem, substitutions are made on the entities, followed by resolution. As shown in Figure 1, cx is a clause from the reasoning clause set S, where cx matches with CX in the rule base, thus replacing the variable X of CX with the entity x of cx, while also calculating the conclusion CY of CX in the rule base to obtain the computational result cy. Cy is the resolvent of cx, and by adding cy to the current reasoning clause set S1, S and S1 are equivalent, indicating that cx has utilized rule CX to perform a reasoning step [10].

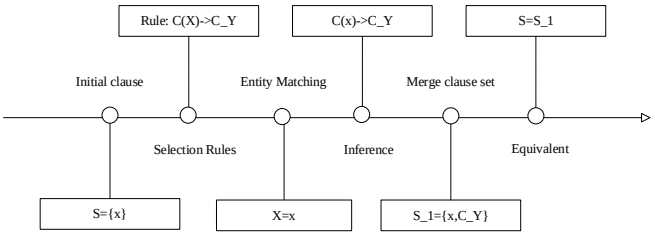


Figure 1. Principle of Resolution

In implementation, a modular design scheme is used to ensure relative independence between each module. The core matching algorithm adopts a hybrid matching mode, combining various matching schemes to accelerate the speed of rule entity matching, forming a mapping table, and ultimately completing the knowledge update.

III. The Overall Framework Design of AORBCO Model Planning System

A. Overall Framework Design

At present, there are 77 high-level strategies, which involve the following issues: remainder, sum multiple, sum difference, difference multiple, division, tree planting, averaging, meeting, two-way, catching up, running water, concentration, profit and loss, lifting, separation movement, chicken and rabbit in the same cage, train crossing the bridge, circular meeting, tax payment and interest, discount and profit, etc [10].

Firstly, a large number of topics are collected, and the high-quality topics are selected. Based on this, the topic data set is expanded, and then they are preprocessed and structured. The knowledge map of mathematical basic rules is established to form the domain knowledge base in AORBCO model, including descriptive knowledge and process knowledge, which paves the way for the subsequent generation of strategic knowledge; The AORBCO model (which consists of belief, ability, desire, planning and execution) plans the matching operation and reasoning calculation process of solving problems, and then discusses the existing cloud computing technology from the perspective of artificial intelligence and epistemology, forming the ability of topic classification and rule selection intelligence in AORBCO model; Finally, relying on the classification of topics in AORBCO model and the ability to select intelligent rules, the limitation of solving existing problems can be

changed through this model, and Ego individuals can master the planning and implementation according to the existing belief knowledge and ability in this process, and obtain the results of new problems that have changed. The system design approach is shown in the following Figure 2.

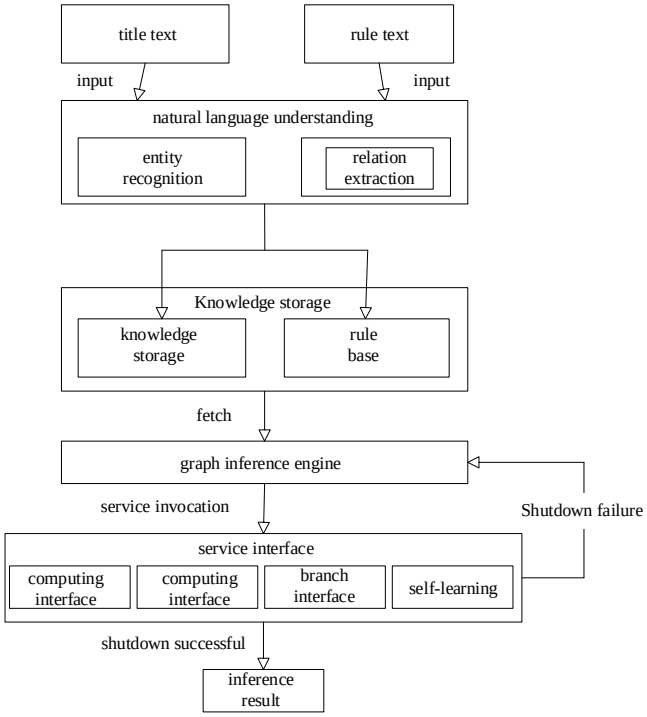


Figure 2. Overall design of the system

B. Self-Learning Mechanism

Self-learning is a branch of machine learning, especially a kind of unsupervised learning problem [11]. The optimization model of self-learning mechanism refers to the design and optimization of self-learning algorithm in multi-agent system by using the principles and methods of game theory, so that each agent can adjust its strategy according to its own goals and changes in the environment, thus achieving a balanced or coordinated state. In the AORBCO model, it refers to its natural learning mechanism, and uses a shallow neural network to calculate the semantic distance between the recognized topic texts.

As the object of matching algorithm of reasoning engine, rule base needs high accuracy. In order to reduce the time, cost of constructing rule base artificially, a self-learning module is added, and relevant rules are extracted from standard answers in a data-driven way, and the processes of reasoning, construction and optimization are automatically carried out, and finally the automatic construction of rule base is completed. Different from the traditional top-down knowledge base construction, the self-learning mode is used to form the rule base from the data from the bottom up, so that the automatic construction ensures the unification of engine rules and reduces the potential problems that may occur in the matching algorithm. In order to ensure the simplicity and unity of the reasoning system, it is necessary to transform the information of the rule subgraph into a data structure and provide it to the matching reasoning module. The process of rule standardization includes initialization rule classification, rule conclusion triplet, rule description, rule knowledge points and other information. Table 1 below is a data structure with structured rules.

TABLE I. RULE STRUCTURING

member name	data structure	describe
label	String	unique identification of the rule
ruleTriple	List<GraphTriple>	regular triplet
conclusionTriples	List<GraphTriple>	rule conclusion triplet
instantiatedCategory	String	Rule classification
instantiatedDescription	String	Simple description of rules
commonText	String	Regular mathematical text

The data of the self-learning system consists of questions and their standard answers, that is, Q=(q,a). Q is the complete question input, A is the complete answer input, and the question and its standard answer are passed into the self-learning module as a set of data. The system will pass the preprocessed result into the inference module, and the input at this time is qt=(qt,at,t), where qt represents the conditional triplet set of the t inference and at represents the result triplet set of the t inference. As a rule, to be evaluated, the results and conditions of reasoning are generated in the generator [12]. Finally, the generated rules are evaluated by the evaluation module, and the rules with high confidence are put into the rule base. The matching process is shown in Figure 3.

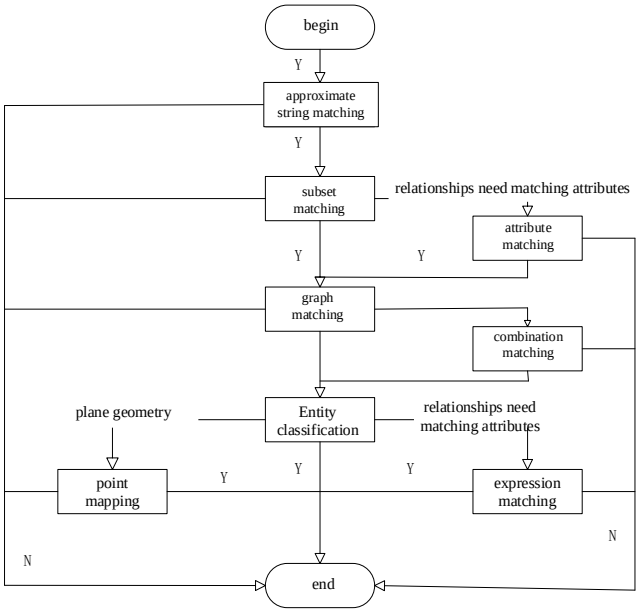


Figure 3. Matching Process

IV. The General Reasoning Design Based on Knowledge Organization

A. Experimental Design

In order to verify the effectiveness and practicality of the automatic problem-solving technology for elementary school mathematics competitions based on the AORBCO model, this study designed a series of experiments. The experimental data is sourced from the NuminaMath-CoT dataset, which contains 860,000 mathematical problems, covering topics from Chinese elementary school mathematics exercises to international mathematical Olympiad questions. To ensure the quality of the problems, this study selected 39,880 questions as the data source and chose 20% of them as the test set. The testing content mainly includes single-instance testing and batch testing.

1) Data Cleaning

Removal of duplicate problems and those with formatting errors.

Tokenization and Part-of-Speech Tagging: Using the HanLP tool for tokenization and part-of-speech tagging of the problems.

Entity Recognition: Identifying mathematical entities in the problems, such as numbers, variables, operators, etc.

Relation Extraction: Extracting mathematical relations from the problems, such as equations, inequalities, functional relationships, etc.

Knowledge Graph Construction: Transforming the extracted entities and relations into nodes and edges in a knowledge graph, stored in the Neo4j graph database.

2) Testing Environment

The software and hardware environment is shown in Table 2 below.

TABLE II. EXPERIMENTAL ENVIRONMENT

Component	Details
Hardware	Intel(R) Core(TM) i7-3770 CPU
	16GB RAM
	1.5T hard disk
Software	Windows 10
	Java development platform IDEA
	Graph database Neo4j
	Symbolic computation platform Maple

3) Single Case Test

The single test aims to verify the system's understanding and problem-solving ability regarding a single question. Representative elementary mathematics problems are selected, and the system is used to solve them, comparing the results with standard answers to assess the accuracy of the solutions and the interpretability of the problem-solving process.

4) Batch Test:

Batch testing is used to evaluate the system's performance on large-scale datasets. A total of 39,880 questions are selected from the NuminaMath-CoT dataset, with 20% designated as the test set, amounting to 7,976 questions. The system automatically solves the problems, and metrics such as the success rate and average solving time are recorded to analyze the overall performance of the system.

B. Testing Results

1) Single Case Test

Testing results indicate that both systems require testing, which is divided into two parts. The first

part is the single test, which begins with the input question text of the problem-solving system. This includes checking whether the functions of each module are complete and whether the modules are interconnected. The following question is selected for the single test: Given that the length of a rectangle is three times its width and the perimeter is 48 centimeters, find the length and width of the rectangle.

The problem-solving process involves understanding the question: the system first processes the question using natural language processing to extract key information: the length of the rectangle is three times its width, and the perimeter is 48 centimeters.

Knowledge graph construction: The extracted information is transformed into nodes and relationships in the knowledge graph, such as the relationship between the length and width of the rectangle and the formula for calculating the perimeter.

Matching reasoning: The system matches corresponding rules based on the information in the knowledge graph, such as the formula for the perimeter of a rectangle P=2(l+w), where l is the length and w is the width.

Parameter reasoning substitution: Based on the conditions in the question, the length is expressed as three times the width, i.e., l=3w. Substituting into the perimeter formula yields 48=2(3w+w).

Calculation: Solving the equation 48=8w gives w=6 centimeters, and subsequently, l=18 centimeters.

Result output: The system outputs that the length of the rectangle is 18 centimeters and the width is 6 centimeters.

As a result, the system successfully solved the problem, and the problem-solving process aligns with the standard answer, taking 30 seconds. The final number of test cases passed by the problem-solving system is shown in Figure 4, with an average time of 1 minute and 20 seconds.

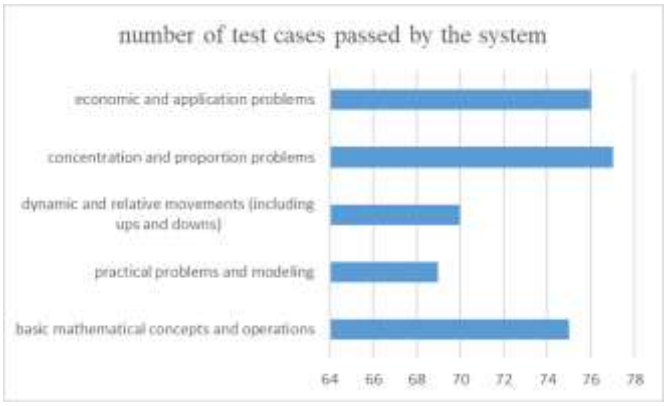


Figure 4. Number of test cases passed by the system

2) Batch Test

The second part is batch testing, which for the inference system mainly includes a total of 500 questions across different modules, covering five common categories: basic mathematical concepts and operations, practical problems and modeling, dynamics and relative motion (including ascent and descent), concentration and ratio problems, and economic and application problems. The primary focus is on assessing the stability of the system. The statistical results of the tests are shown in Figure 5.

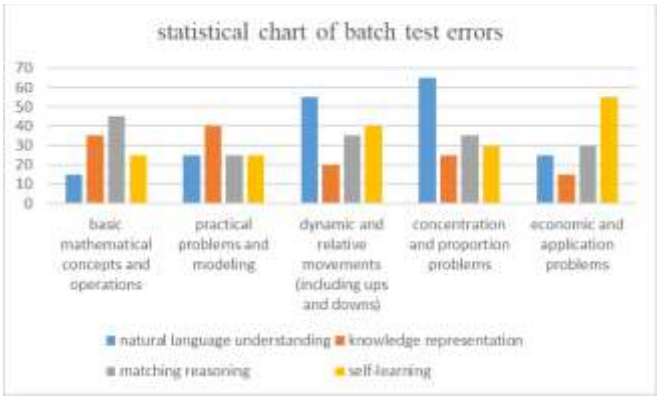


Figure 5. Statistical chart of batch test errors

The success rate of problem-solving: The system successfully solved 6260 out of 7976 problems, resulting in a success rate of 78.5%. This indicates that the system performs well in handling the majority of elementary school mathematics competition problems, but there is still room for improvement.

Average problem-solving time: The average problem-solving time is 1 minute and 30 seconds, which is acceptable in actual teaching and learning

scenarios. However, for some complex problems, the solving time is longer and requires further optimization. The optimized second-phase system has shown certain improvements in various modules compared to the first phase, as illustrated in Figure 6.

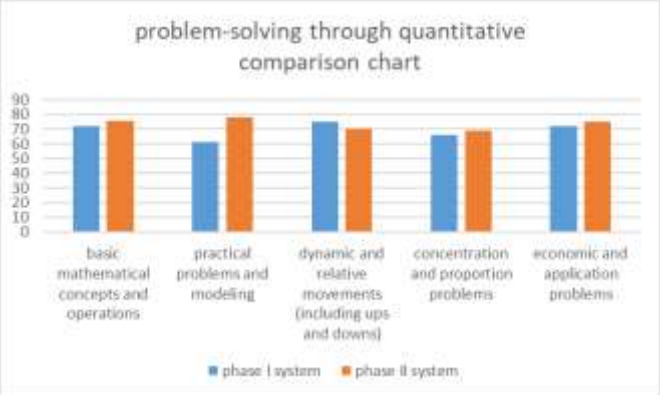


Figure 6. Problem-solving through quantitative comparison chart

3) Comparative Testing

In order to further evaluate the performance of the system, a comparative test will be conducted between this system and existing elementary school mathematics problem-solving software (such as Xiaoyuan Search Questions and Homework Help). A selection of 500 representative problems will be used to solve using these software, and the success rate and average solving time will be recorded.

The comparison results are shown in Table 3:

TABLE III. THE COMPARISON OF THE PROBLEM-SOLVING SUCCESS RATES BETWEEN THIS SYSTEM AND OTHER PLATFORMS

Problem-solving system	Success rate of problem-solving	Average problem-solving time
This system	78.5%	1min30s
Little ape search questions	65%	2min10s
Homework Help	60%	2min30s

C. Result Analysis

In the batch testing, the system automatically solved 7976 questions, achieving a success rate of 78.5%. The following Table 4 provides a detailed analysis of the success rates and average solving times for different types of questions:

TABLE IV. COMPARISON OF PROBLEM-SOLVING EFFECTIVENESS ACROSS DIFFERENT QUESTION TYPES

Type of question category	Success rate of problem-solving	Average problem-solving time
Basic Operations and Relations	85%	1min10s
Geometry and tree planting	75%	1min30s
Application problems	72%	1min40s
Special question types and techniques	68%	1min50s
Other categories	80%	1min20s

It can be seen from the above table that the system has the highest success rate in solving basic operations and relational problems, with the shortest average solving time. In contrast, the success rate for special types of questions and skill-based problems is the lowest, with the longest average solving time. This indicates that there is still room for improvement in the system's handling of complex problem types.

Success Rate: This system significantly outperforms Xiaoyuan Search and Homework Help in terms of success rate, improving by 13.5% and 18.5% respectively. This indicates that this system has higher accuracy and reliability when dealing with elementary school mathematics Olympiad problems.

Average Solving Time: The average solving time of this system also surpasses that of Xiaoyuan Search and Homework Help, reducing by 40 seconds and 1 minute respectively. This indicates that this system also has a significant advantage in solving efficiency.

To demonstrate the dynamic characteristics of knowledge weights during the reasoning process, we conducted feature tracking experiments on 500 problem-solving cases. As shown in Figure 7, the knowledge weight (calculated by Formula 1) shows an exponential decay trend during the initial reasoning phase (0-30s), but exhibits periodic reinforcement patterns after rule matching and cognitive optimization modules are activated. Notably, when the reasoning path encounters dead

ends (marked by red arrows), the system triggers backtracking mechanisms that significantly enhance the weights of alternative knowledge nodes (average +23.6%).

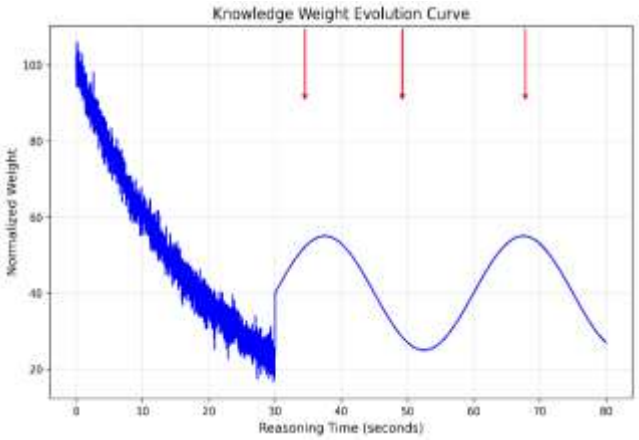


Figure 7. Knowledge weight evolution during problem-solving process

D. Summary of Test Results

The "Chicken-Rabbit Cage Problem" was selected for its multi-path solution characteristics (algebraic, enumerative, and substitution methods), moderate reasoning depth (average 6.8 steps), and explicit intermediate variable requirements.

Figure 8 illustrates the phased analysis using a dual-axis timeline (10ms sampling resolution). The primary axis tracks active knowledge nodes (weight threshold θ =40), while the secondary axis monitors weight concentration dynamics. Four distinct phases emerge:

Knowledge Activation (0-5s): Initial filtering reduced active nodes from 12→9.

Rule Matching (5-25s): Constraint identification increased weight concentration from 54%→61%.

Cognitive Optimization (25-35s): Path pruning (58→16 paths) boosted concentration to 89%.

Convergence Verification (>35s): Final validation through algebraic proof.

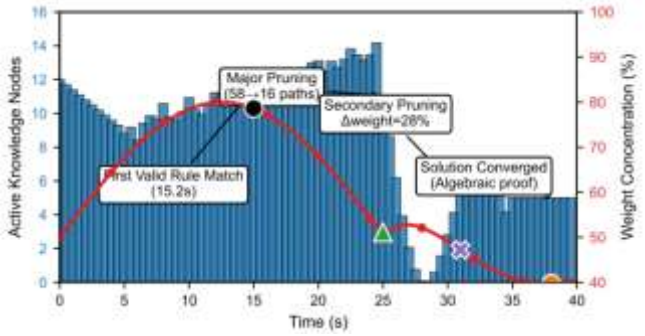


Figure 8. Temporal evolution of active nodes (bars) and weight concentration (line) during problem solving

The system demonstrated 72.4% search space reduction through three optimization waves (Table 6). Error recovery analysis revealed 2.4s mean detection latency for pseudo-solutions, with backtracking depth of 2.3 steps to valid checkpoints.

TABLE V. PHASE TRANSITION PARAMETERS

Phase	Active Nodes	Weight Concentration	Trigger Condition
Initial Activation	12→9	N/A	Knowledge filtering
Rule Matching	9→14	54%→61%	Constraint identification
Cognitive Optimization	14→5	61%→89%	Path pruning activation

Through single-instance testing and batch testing, this system has demonstrated excellent performance in both solving accuracy and efficiency. In single-instance testing, the system successfully solved the problem, with the solving process consistent with the standard answer, taking 30 seconds. In batch testing, the system successfully solved 6260 out of 7976 problems, achieving a success rate of 78.5% and an average solving time of 1 minute and 30 seconds. In comparative testing, this system outperformed existing elementary school mathematics Olympiad solving software in both success rate and average solving time.

Although the system performed excellently in testing, there are still areas that require improvement. For instance, the system takes longer to solve some complex problems, necessitating further optimization of the matching algorithm and reasoning engine. Additionally, the system still

makes errors when handling certain special problem types, requiring further expansion of the rule library and optimization of the self-learning module.

From the statistical chart of error situations in various modules of batch testing, it can be seen that the four modules of the system: natural language understanding, knowledge representation, reasoning system, and self-learning exhibit varying pass rates across different problem types. Due to the nature of the problem type, all modules performed poorly on sequences, while planar geometry faced significant issues in natural language understanding due to its complex expressions and multiple references.

The number of rules generated by the self-learning module is positively correlated with the pass rate of the solving system tests. For different modules, self-learning is also related to the performance of the natural language understanding module. The understanding of standard answers affects data quality, and the performance of the reasoning system impacts the rule merging part of the automatically generated rules, resulting in a high number of rules that cannot be merged, leading to insufficient data volume for the system's reasoning results.

Single-instance testing has proven the completeness of the functions of each module of the solving system, and the statistical results of batch testing also reflect a high degree of connectivity among the system's modules. The system has achieved the basic functions specified in the initial phase, with an average solving rate of 73.4%.

E. Rule Base Growth Pattern

The self-learning module's performance was quantified through continuous 72-hour operation monitoring. As shown in Table 5, the rule base demonstrates logarithmic growth characteristics, with rule generation speed decreasing from 12.5 rules/hour to 4.2 rules/hour as system maturity increases. The error rate of automatically generated rules shows strong negative correlation (r=-0.87) with the accumulated rule quantity.

TABLE VI. RULE BASE EVOLUTION METRICS

Time Interval (h)	New Rules Generated	Error Rate (%)	Avg. Confidence
0-12	148	18.2	0.76
12-24	92	12.1	0.83
24-48	165	9.7	0.88
48-72	101	6.3	0.91

V. CONCLUSIONS

The non-linear growth pattern of rule base suggests that the system follows similar learning curves to human students, where initial rapid knowledge acquisition gradually transitions to refinement optimization. The observed 62.4% error reduction rate during the first 24 hours demonstrates the effectiveness of our cognitive optimization algorithms.

The reasoning engine, as the core of the problem-solving system, employs traditional forward reasoning methods and integrates computation during the reasoning process, continuously iterating to generate new knowledge. In terms of implementation, a modular design scheme is utilized to ensure relative independence among each module. The core matching algorithm adopts a hybrid matching mode, combining various matching schemes to accelerate the speed of rule entity matching, forming a mapping table, and ultimately completing the knowledge update.

Currently, if the reasoning module in the problem-solving system fails to successfully comprehend the entity information within the dataset, the method anticipated to improve Ego's accuracy in determining the processing requirements of the dataset tasks is as follows: if this type of problem requirement cannot be understood temporarily in Ego's knowledge base, user input can be utilized to enhance Ego's semantic recognition of the requirement through natural language processing, thereby providing a specific problem-solving method tailored to this requirement; additionally, there is the issue of information loss caused by the matching algorithm. In this case, a method using associated nodes is adopted, establishing a logically equivalent relationship between the nodes before and after the update, treating the two nodes as the same entity

during use.

The elementary school mathematics automatic problem-solving system based on the AORBCO model has achieved significant results in both problem-solving accuracy and efficiency, providing strong technical support for elementary school mathematics education. Future research will focus on further enhancing the model's generalization ability, exploring the integration of multimodal learning, and developing a more intelligent personalized learning tutoring system.

REFERENCES

[1] Cui Dan, Li Shuqi. Design of natural language information extraction-translation-proofreading system based on AI algorithm. Modern Electronic Technology, 2024, 47(10):111-116.

[2] Chen Yuncai. Research on Natural Language Processing Technology Based on Artificial Neural Network. Engineering Technology Research, 2024,9(08):93-95.

[3] Li Bo. Application of Deep Learning in Natural Language Processing. Electronic Technology, 2024, 53(04):425-427.

[4] Feng Shaoxian. Research on OCR detection and recognition technology based on deep learning. north china university of technology, 2023.

[5] Feng C, Lu L, Gao W. Research and Design of Planning Systems in the AORBCO Model. International Journal of Advanced Network, Monitoring and Controls, 2023, 8 (3): 1-9.

[6] Hanpeng L, Wuqi G, Junmin L. Research on Intelligentization of Cloud Computing Programs Based on Self-awareness. International Journal of Advanced Network, Monitoring and Controls, 2023, 8 (2): 89-98.

[7] Jiang Haitao. Research on key technologies and system implementation of automatic problem-solving algorithm for mathematical application problems. Donghua University, 2023.

[8] Dou Ruolin. Realization of automatic problem-solving and problem-setting system based on mathematical semantic understanding. Beijing University of Posts and Telecommunications, 2023.

[9] Xiao Liangshun. Research on knowledge fusion in AORBCO model. Xi'an University of Technology, 2023.

[10] Liu Ben. Research on Natural Language Understanding in AORBCO Model. Xi'an University of Technology, 2023.

[11] Guo Runze. Design of natural learning mechanism in AORBCO model. Xi'an University of Technology, 2021.

[12] Wang Ziyun. Research and application of semantic enhancement technology in natural language understanding of elementary mathematics based on knowledge map. University of Electronic Science and Technology of China, 2023.

Research on Early Prediction of Lung Cancer Based on Deep Learning

Zhijun Qu
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: quzj158@163.com

Zhongsheng Wang
State and Provincial Joint Engineering Lab. of Advanced Network, Monitoring and Control
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: wzhsh1681@163.com

***Abstract*—Cancer of the lung is a principal cause of mortality due to cancer on a global scale. Traditional imaging techniques suffer from subjectivity limitations. Meanwhile, convolutional neural networks (CNNs) within deep learning, though highly effective in image classification, still have limitations when dealing with complex and data-scarce medical images. To address this challenge, this paper proposes a data-efficient image Transformer (DeiT) model based on the Transformer architecture with a self-attention mechanism, enhanced through knowledge distillation. This model can capture global information in images and improve the classification accuracy of lung cancer images under small-sample conditions by leveraging a teacher model. Through model training and evaluation, results demonstrate that the DeiT model achieves an impressive prediction accuracy of 99.96% under small-sample medical imaging conditions. This highlights the advantages of the Transformer architecture in medical image analysis. The findings provide a new perspective for early lung cancer detection and underscore the powerful performance of the DeiT model in handling complex small-sample data conditions.**

Keywords-component; Lung Cancer Detection; Deep Learning; Knowledge Distillation; DeiT Model; Medical Image Analysis; Small-Sample Learning

I. INTRODUCTION

Over the past few years, AI's dramatic progression has spurred substantial breakthroughs in deep learning (DL), especially in computer vision. Convolutional neural networks (CNNs) are widely used in image classification and object detection. These models have found widespread applications in domains like facial recognition and autonomous driving, and security surveillance due to their powerful feature extraction capabilities. These advancements have improved image processing efficiency and accelerated the development of intelligent healthcare, making medical image analysis a key application of deep learning [1].

In the medical domain, particularly in tumor diagnosis and early detection, the analysis of medical imaging data poses significant challenges. Traditional imaging techniques like X-rays, CT, and MRI rely on physicians' expertise. However, the vast amount of image data and the complex nature of lesion morphology make these methods vulnerable to human error, increasing the chances of misdiagnosis or missed diagnosis. Lung cancer continues to be a leading cause of mortality globally, with early detection being essential for enhancing survival rates. However, early-stage lung cancer presents subtle symptoms, and its imaging data is complex, making traditional methods insufficient for efficient and accurate detection. Pathological image analysis depends on manual interpretation by pathologists, but due to the intricate details of tissue slices, this process is time-consuming and prone to errors, especially when detecting subtle cellular changes.

Deep learning has demonstrated great potential in medical image analysis. Conventional convolutional neural networks (CNNs), like VGG and ResNet, have attained remarkable outcomes in image classification and object detection through the incorporation of deeper network architectures and residual connections. However, these traditional CNNs typically focus on local features, making it difficult to effectively capture global

contextual information. This limitation is particularly evident when processing complex medical images, as local features may not fully represent the true nature of the disease. Additionally, medical image resources are often scarce, which presents another challenge. In this scenario, DeiT, a data-efficient image processing model that utilizes the Transformer architecture, has been recognized as a notable development in deep learning research in recent years.

The DeiT model not only captures global information from images through a self-attention mechanism but also enhances the ability to learn from small sample data through knowledge distillation, leveraging powerful teacher models. Compared to traditional CNN models, DeiT has an advantage when processing limited medical image data and has shown excellent performance in tasks such as early lung cancer detection.

This paper aims to explore the application of the DeiT model in early lung cancer detection. By analyzing lung pathological images, this paper compares the performance of DeiT with traditional convolutional neural networks (such as VGG and ResNet) and evaluates its accuracy and potential applications in lung cancer detection. Through experiments and data analysis, this paper aims to validate the advantages of the DeiT model under the Transformer architecture in medical image analysis, particularly under conditions of limited data samples, they provide innovative insights and solutions for early lung cancer detection [2].

II. Related Work

Detecting lung cancer at an early stage is essential for lowering its high mortality rate. Conventional imaging diagnostic techniques, including CT scans, X-rays, and pathological image analysis, depend on the expertise of radiologists, which can introduce subjectivity and potential misdiagnosis. With the progress of computer vision and deep learning, image-based lung cancer detection methods have attracted significant attention and are increasingly being applied in medical diagnosis.

A. Lung Cancer Detection Based on Deep Learning

Lately, approaches rooted in deep learning, most notably Convolutional Neural Networks (CNNs), has exhibited outstanding performance in medical image analysis and has progressively taken the place of traditional feature extraction approaches. Investigations reveal that CNNs have achieved remarkable success in identifying lung cancer images. Numerous studies have used deep learning models for CT screening, yielding high accuracy and sensitivity. Additionally, Cohen et al. developed an automated lung cancer detection system by applying deep learning to analyze lung nodules, surpassing the average performance of radiologists. In China, deep learning has also been widely applied. For example, Li Ming et al. used an improved ResNet model to classify lung cancer CT images, achieving high accuracy. Meanwhile, Zhang Hua et al. proposed a lung cancer detection method that integrates multiple deep learning networks, further enhancing the model's detection capabilities[3].

B. Lung Cancer Detection Based on Pathological Images

Unlike CT images, pathological images provide higher resolution, cell-level images, making them of significant value in cancer detection and diagnosis. Recently, there has been a rising focus on using pathological images for lung cancer detection. For example, Liu et al. developed a CNN-based technique to classify lung cancer pathological images, effectively distinguishing between malignant and normal regions. The research shows that, despite the relative scarcity of pathological image data, deep learning models, especially CNN-based architectures, can still achieve good classification results when processing this high-resolution data[4].

C. Knowledge Distillation and Small-Sample Learning in Lung Cancer Detection

With the continuous development of deep learning technology, knowledge distillation and few-shot learning have gradually become new directions in lung cancer detection. In recent years, researchers have focused on leveraging knowledge

distillation techniques to transfer insights from larger models to smaller ones, with the goal of improving the efficiency of lung cancer detection[5]. Studies have shown that the DeiT model excels in handling rare data, especially in the field of medical imaging, demonstrating strong generalization ability.

Therefore, deep learning-based lung cancer detection methods significantly improve accuracy and efficiency compared to traditional methods, but still face challenges such as data scarcity and environmental complexity. The lung cancer detection method based on the DeiT model proposed in this paper aims to improve the accuracy of lung cancer detection under few-shot conditions[6].

III. Technical Model

A. Convolutional Neural Networks (CNN)

Convolutional Neural Networks (CNNs), as the foundation of the two traditional models in the comparative experiments of this paper, utilize convolution operations to extract image features and combine mechanisms such as pooling to reduce data dimensions, thereby optimizing the processing efficiency of high-dimensional visual data. In this paper, two classic representative CNN models, VGG16 and ResNet50, were selected to train and test the early lung cancer detection task, in order to explore their performance in classification on complex medical small-sample image datasets. Figure 1 presents the fundamental structure of the CNN.

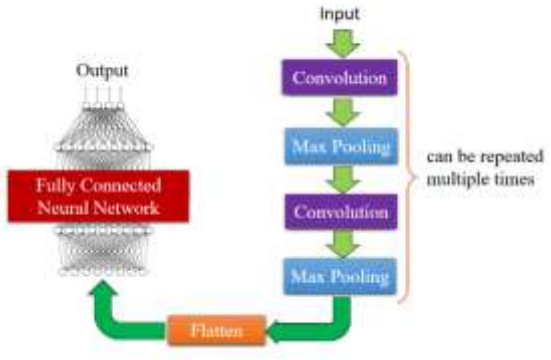


Figure 1. The basic architecture of a convolutional neural network

1) Convolution and Pooling

For Convolutional Neural Networks (CNNs), convolution acts as an essential process to extract features from the input images. This process employs a small convolutional kernel, such as a 3x3 or 5x5 matrix, to produce a feature map from the input image. The procedure entails an input image matrix I (with dimensions HxW) and a kernel matrix K (with dimensions kxk), along with a stride S. To control the size of the feature map, padding can be applied. Typical padding techniques include "valid padding" and "same padding," ensuring the output dimensions match the input image. The mathematical formulation is shown in Equation (1).

$$O(i,j)=\sum_{m=0}^{k-1}\sum_{n=0}^{k-1} I(i+m,j+n)\cdot K(m,n) \quad (1)$$

The element situated at position (i,j) within the output feature map is represented as O(i,j). I(i+m, j+n) is the corresponding element in the input image that aligns with the convolution kernel. K(m,n) represents the elements of the convolution kernel. The variable k represents the size of the convolution kernel. The dimensions of the feature map can be determined using the following equations (2) and (3).

$$\text{Output\_Height}=\frac{H-k+2P}{s}+1 \quad (2)$$

$$\text{Output\_Width}=\frac{W-k+2P}{s}+1 \quad (3)$$

Here, P represents the number of padding pixels (additional pixels at the image edges), and s is the stride. The convolution process is illustrated in Figure 2.

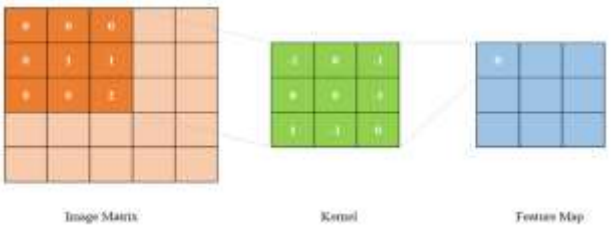


Figure 2. Convolution Process Diagram

The pooling layer in CNNs down samples feature maps, reducing dimensionality and computational complexity while enhancing translation invariance and preventing overfitting. Typically placed after the convolutional layer, it helps the model focus on essential image features. Max pooling is a widely used technique that selects the maximum value from a 2×2 or 3×3 window, typically with the stride equal to the window size. The formula is given in Equation (4).

$$O(i,j)=\frac{1}{k^2}\max_{m=0}^{k-1}\max_{n=0}^{k-1} I(i+m,j+n) \qquad (4)$$

Here, O(i,j) designates the element located at the coordinates (i,j) in the output feature map, and k represents the size of the pooling window. Figure 3 depicts the pooling operation.

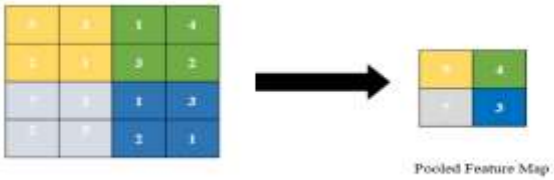


Figure 3. Pooling Process Diagram

2) VGG16 Model

VGG-16 is composed of 13 convolutional layers, each employing a 3×3 kernel, with a ReLU activation function applied after each convolution to maintain the network's non-linearity. After multiple convolutional layers, a max pooling layer is incorporated to execute down-sampling and decrease the feature dimensionality [7]. The entire model includes 5 pooling layers.

Following the convolutional and pooling layers, VGG-16 consists of three fully connected layers. The first two layers contain 4096 nodes each, while the final fully connected layer produces the classification results. The last component of the network is a Softmax output layer, which transforms the outputs of the fully connected layers into a probability distribution, indicating the probability that the image falls into each category. This architecture is depicted in Figure 4.

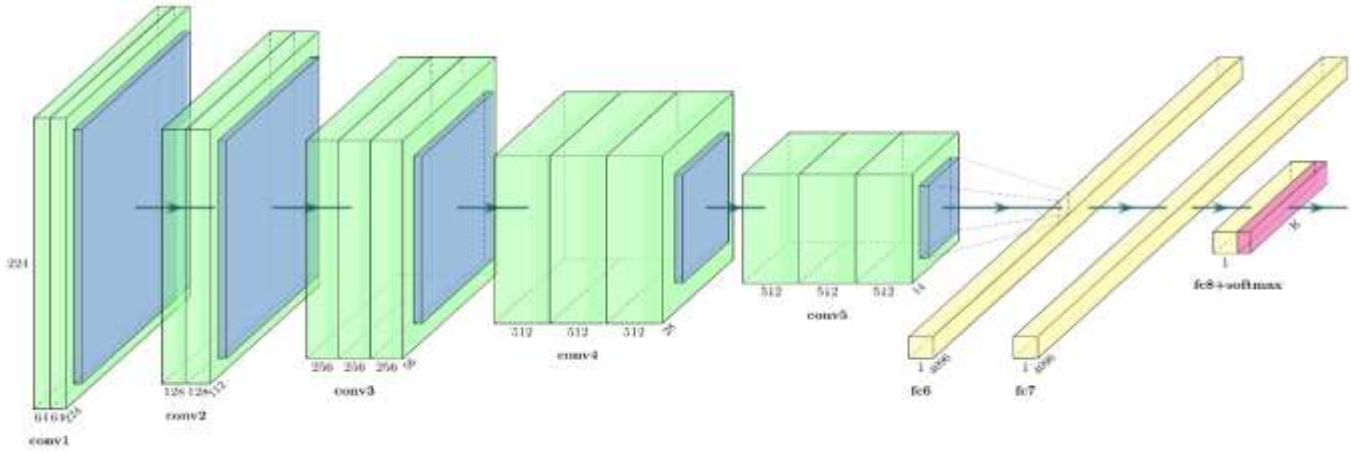


Figure 4. VGG16 Network Architecture Diagram

3) ResNet Model

Research on Convolutional Neural Networks (CNNs) has demonstrated that as deep neural networks grow deeper, they often encounter the problems of vanishing or exploding gradients during the training phase, thereby making training more challenging. In order to address this issue, the ResNet [8] model proposes a residual learning framework, which effectively overcomes the challenges associated with training deep networks.

Residual learning relies on skip connections, allowing the input to bypass layers and pass

directly to later ones. This helps the network learn a residual function instead of directly learning the mapping function. By optimizing the residual function, the network can capture complex features more effectively. The purpose of residual learning can be described by Equation (5).

$$y = F(x,\{Wi\}) + x \tag{5}$$

Assume x indicates the input, while F(x,{Wi}) illustrates the residual function, made up of nonlinear transformations (generally two or three convolutional layers) with parameters Wi. Let y denote the output. The input x is directly transmitted to the output via a skip connection, subsequently being merged with the result of F(x,{Wi}). This procedure is depicted in Figure 5.

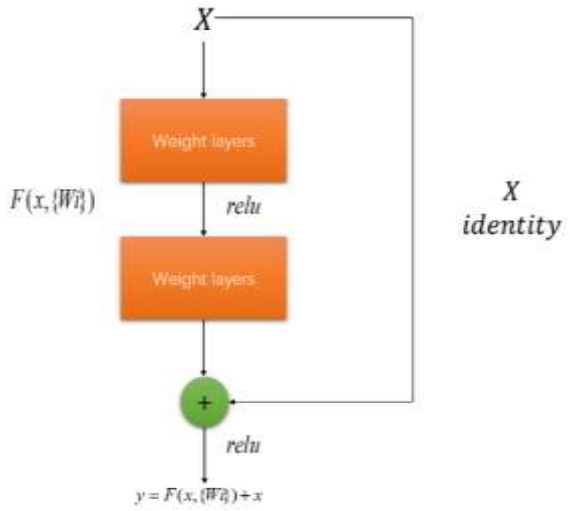


Figure 5. Residual Connection Structure Diagram

B. Transformer Model

The Transformer model employs the attention mechanism and differs from the architectures of Recurrent Neural Networks (RNN) and Long Short-Term Memory networks (LSTM) This design enables it to excel in parallel computation and effectively capture long-range dependencies. Since its introduction, the Transformer model has gained prominence, particularly in NLP and Computer Vision. Its architecture is illustrated in Figure 6.

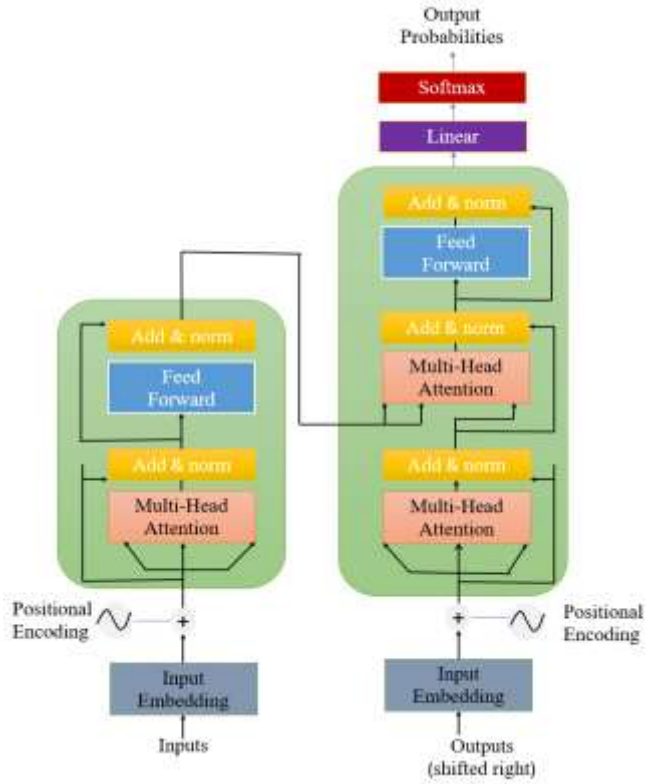


Figure 6. Transformer Architecture Diagram

1) Self-Attention Mechanism

The self-attention mechanism is an essential approach in deep learning for managing sequential data, and it is widely utilized in Natural Language Processing (NLP) and Computer Vision (CV). The core idea is to consider the influence of other parts when processing each part of the input data, allowing the model to capture relationships between different parts and helping to capture long-range dependencies, thereby improving performance on long sequences. The operation principle is as follows: Given an input sequence X = {x_1, x_2,..., x_n}, K denotes the key vectors, while V represents the value vectors. The similarity between each query vector Q and all the other key vectors K is determined by computing their dot product, then the results are normalized with a softmax operation to obtain the attention weights for each element. Equation (6) presents the computational formula.

$$Attention_{ij} = \frac{exp(Q_i \cdot K_k)}{\sum_{k=1}^{n} exp(Q_i \cdot K_k)} \quad (6)$$

Each element's representation is updated by computing a weighted sum of all value vectors Vj, where the weights are determined by the attention weights computed above. The calculation formula is provided in Equation (7). Eventually, the final result comprises a series of representations derived from this weighted sum.

$$Output_i = \sum_{j=1}^{n} Attention_{ij} \cdot V_j \quad (7)$$

Unlike traditional RNNs or LSTMs, the self-attention mechanism considers all elements of the sequence simultaneously, allowing better capture of long-range dependencies. Since it does not rely on the order of input elements, computations can be parallelized, improving efficiency. It can be applied to various data types, including text and images. Self-attention is a powerful technique for handling sequential data by modeling similarities between elements, enabling better understanding of internal relationships. It is widely used in advanced models like Transformer, ViT, and DeiT, offering significant advantages, especially for long sequences.

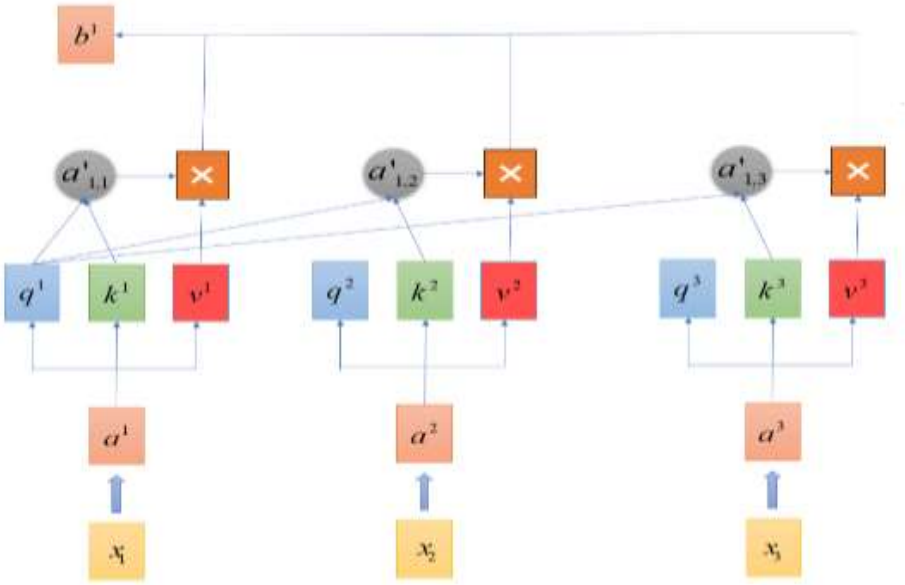


Figure 7. Self-Attention Mechanism Computation Diagram

2) ViT(Vision Transformer) Model

The Vision Transformer (ViT) is an image classification model that is founded on the Transformer architecture. It splits an image into fixed-size patches, treating each as a "token," and uses the self-attention mechanism to process the image, bypassing the convolutional layers typically found in traditional CNNs. ViT captures relationships between distant pixels in an image through self-attention, enabling global modeling[9]. Unlike CNNs, ViT performs better on large-scale datasets and can surpass traditional CNN models on massive datasets like ImageNet.

In ViT, the image is split into several patches, with each patch being transformed into a representation akin to word vectors. Positional encoding is incorporated to allow the model to detect spatial relationships among patches. The processed patches are fed into the Transformer encoder, where their representations are refined through self-attention layers. In the end, the output is categorized through a fully connected layer. Figure 8 illustrates the model architecture.

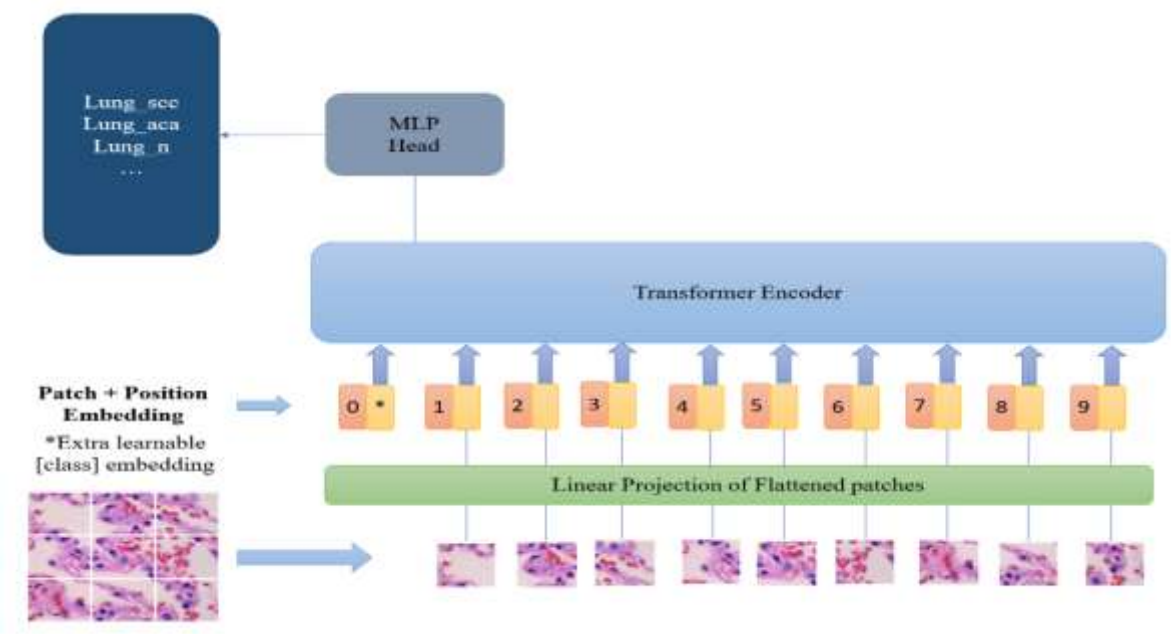


Figure 8. ViT Architecture Diagram

3) DeiT(Data-efficient Image Transformer) Model

DeiT is a variant of ViT specifically designed to improve the data efficiency of Transformer models in image classification tasks. By introducing knowledge distillation, the DeiT model successfully overcomes the inefficiency of training ViT on small datasets, this allows ViT models to perform on par with Convolutional Neural Networks (CNNs), even when there is a limited amount of data [10].

The core design principle of DeiT is based on knowledge distillation. By incorporating a teacher model, knowledge is transferred to the student model (DeiT), allowing it to learn more efficient feature representations, even with a small amount of data. This allows DeiT to achieve strong performance on small datasets while avoiding overfitting or underfitting issues commonly seen in ViT training.

The working principle of DeiT is similar to that of ViT. DeiT first divides the input image into fixed-size patches, flattens each patch, and maps them into an embedding space through a linear transformation. Each patch embedding is then enhanced with positional encoding to preserve spatial information. After positional encoding, the patches are fed into a Transformer encoder, which utilizes self-attention mechanisms to capture relationships between different patches in the image.[11]

The key innovation of DeiT lies in the introduction of the knowledge distillation mechanism to improve the training process. During training, DeiT optimizes two objectives simultaneously: Supervised loss: Computed by comparing the output of the class token with the hard labels. Distillation loss: Computed by comparing the output of the distill token with the soft labels generated by the teacher model.

Specifically, the teacher model generates a probability distribution (soft labels) for each class, capturing the similarities between categories. The distillation loss is calculated using Kullback-Leibler (KL) divergence, which estimates the divergence between the output of the student model's distill token and the soft labels generated by the teacher model [12].

To integrate these two optimization objectives, DeiT defines a total loss function (Ltotal) that combines both the supervised loss and the distillation loss. The formula is provided in Equation (8), where α and β are key weights that balance the two loss components. Here, Ltotal

denotes the total loss, and Lsupervised refers to the supervised loss, and Ldistillation is the distillation loss[13].

$$L_{total} = \alpha L_{supervised} + \beta L_{distillation} \quad (8)$$

The supervised loss Lsupervised optimizes the parameters of the class token to improve classification accuracy, while the distillation loss Ldistillation optimizes the parameters of the distill token, enabling it to learn deep feature representations from the teacher model. Additionally, both loss terms jointly optimize the shared parameters of the Transformer encoder. The distill token is a crucial innovation in DeiT, offering a pathway for the student model to receive knowledge from the teacher model, thereby significantly enhancing its performance on small datasets. Through its innovative design and distillation techniques, DeiT represents a major breakthrough in deep learning for computer vision, demonstrating its strong potential, particularly in image classification tasks. Its model architecture is shown in Figure 9.

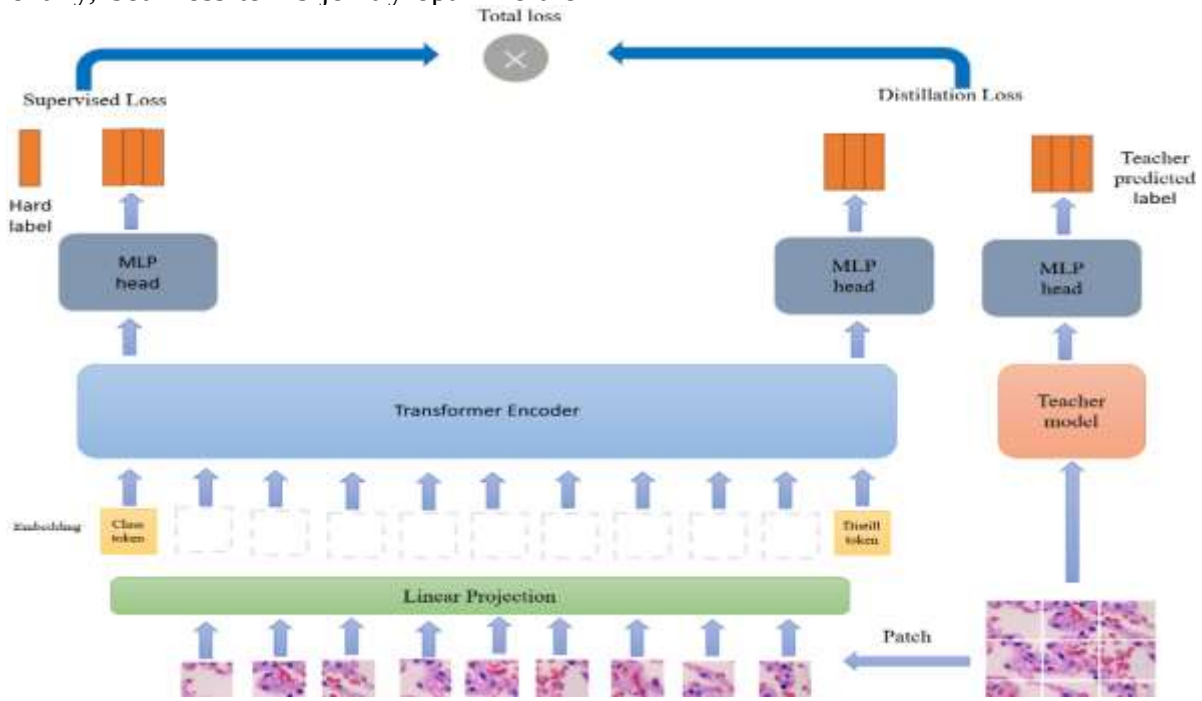


Figure 9. DeiT Architecture Diagram

IV. Experiment And Analysis

A. Experimental Environment and Model Parameters

The models utilized in this experiment were trained and fine-tuned on Kaggle with the help of the TensorFlow and PyTorch frameworks. The code was developed and run in a Jupyter Notebook environment. The hardware configuration included a GPU P100, TensorFlow version 2.16.1, Python version 3.10.14, and PyTorch version 2.4.0. The Adam optimizer, with a learning rate of 0.00001 and cross-entropy loss, was applied during training. A batch size of 64 was used, and the model was trained for 30 epochs. VGG16, ResNet50, and DeiT were evaluated on datasets containing lung adenocarcinoma, lung squamous cell carcinoma, and normal lung tissue.

B. Experimental Procedure

The lung cancer image dataset used in this study consists of 15,000 pathological images, categorized into three groups: lung_aca (lung adenocarcinoma), lung_n (healthy lung tissue), as shown in Table 1, the dataset is divided into a

training set with 10,500 images, a validation set containing 2,250 images, and a test set of 2,250 images.

TABLE I. EXPERIMENTAL DATASET TABLE

	Training Set	Validation Set	Test Set
lung_aca	3500	750	750
lung_scc	3500	750	750
lung_n	3500	750	750

In order to improve the model's capacity for generalization, a variety of data augmentation methods were utilized on the images throughout the training process. These methods included arbitrary horizontal flipping, rotation, and translation shear. The images were also resized to 224x224 pixels and normalized using the mean and standard deviation from the ImageNet dataset, with values (mean = [0.485, 0.456, 0.406] and standard deviation = [0.229, 0.224, 0.225]). Data augmentation increased the diversity of the training data, helping the model learn more transformed features, thereby improving its performance in various scenarios. The images underwent random horizontal flipping with a probability of 10%, while the rotation range was set between -10 and +10 degrees. Random cropping was applied to adjust the images to the target size (224x224), with a cropping ratio ranging from 90% to 110%. Additionally, random transformations through translation and shear were applied, with a maximum transformation of 10%, helping the model adapt to different perspectives and spatial positions. The illustration in Figure 10 displays nine arbitrarily chosen enhanced images derived from the lung cancer pathological training dataset. The images are equally distributed across three different types of lung cancer: lung adenocarcinoma, normal lung tissue, and lung squamous cell carcinoma, as mentioned in [14].

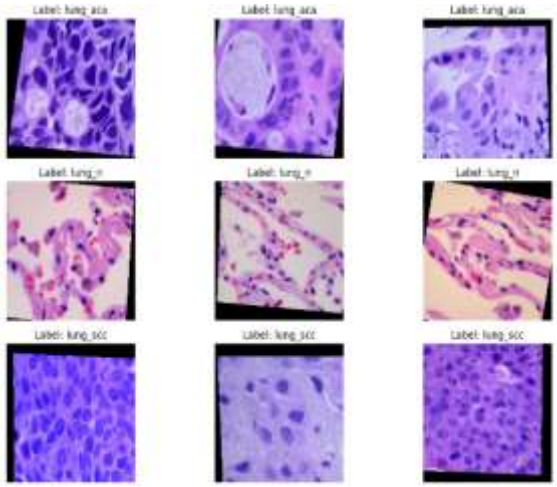


Figure 10. Randomly selected training sample images

In this study, the DeiT model (deit_base_patch16_224) based on Vision Transformer (ViT) was used and compared with traditional convolutional neural network (CNN) models, specifically VGG16 and ResNet50. During training, the Cross Entropy Loss function was used to calculate the loss, and the Adam optimizer was applied to adjust the model's parameters. A learning rate of 1e-5 was chosen, and the training spanned 30 epochs. At each epoch, the loss and accuracy for both the training and validation datasets were calculated to monitor the model's progress.

C. Experimental Results and Analysis

1) Training Results Analysis

The model's stability and ongoing performance improvement were demonstrated by plotting the loss and accuracy curves for both training and validation. As shown in Figures 11 and 12, the accuracy and loss curves for the VGG16 and ResNet50 models exhibited distinct patterns after 30 epochs of training. During the early epochs, the models had lower accuracy and higher loss values. However, as training progressed, the accuracy gradually improved, and the loss decreased. By the end of 30 epochs, both the accuracy and loss curves had stabilized, indicating convergence. Ultimately, the accuracy of VGG16 and ResNet50 reached 98.49% and 97.51%, respectively[15].

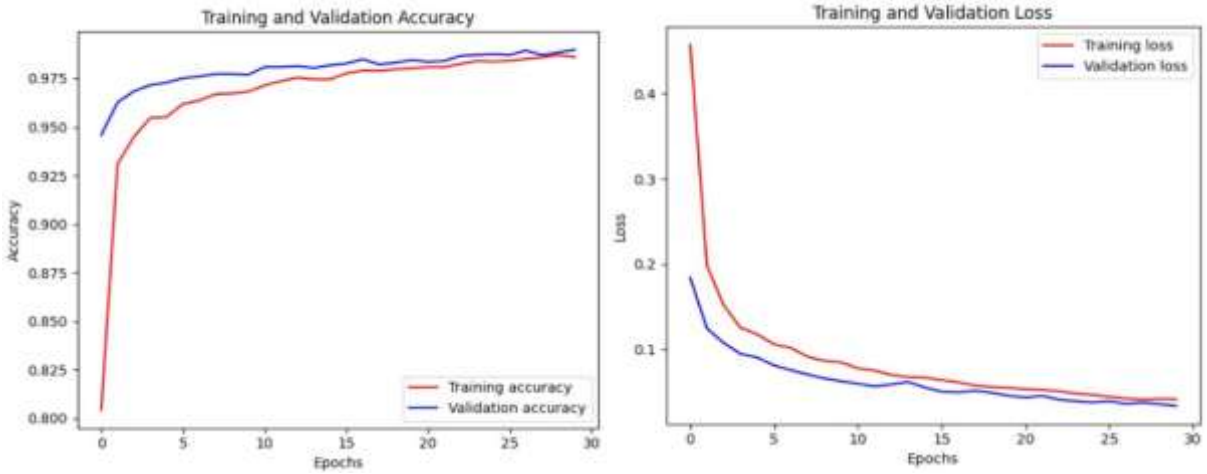


Figure 11. Accuracy and Loss Curves of theVgg16

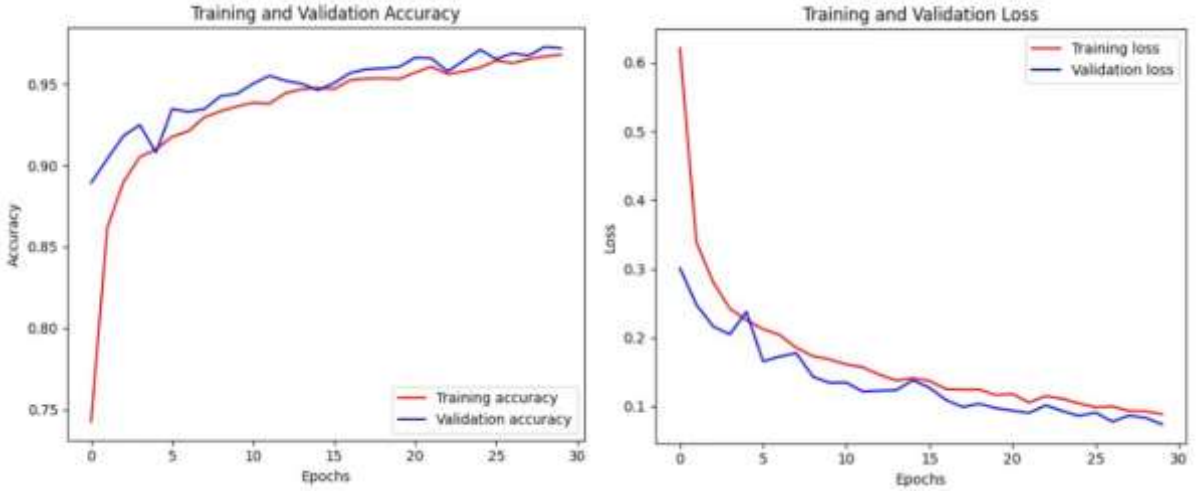


Figure 12. Accuracy and Loss Curves of the ResNet50

The DeiT model achieved the best training results and the highest accuracy, as shown in Figure 13. At the beginning of training, the DeiT model already had a high initial accuracy. This is due to the introduction of knowledge distillation and the self-attention mechanism, which enhance its ability to capture contextual information and extract complex medical image features more effectively. Additionally, the student model benefits from the guidance of the teacher model, allowing it to develop strong feature learning capabilities from the very start. After 30 epochs, the DeiT model successfully converged, achieving an accuracy of 99.96%, demonstrating its outstanding performance in medical image classification tasks. Compared to traditional CNN models, the DeiT model provides higher accuracy on small sample datasets, fully showcasing the powerful advantages of self-attention mechanisms and knowledge distillation in image classification.

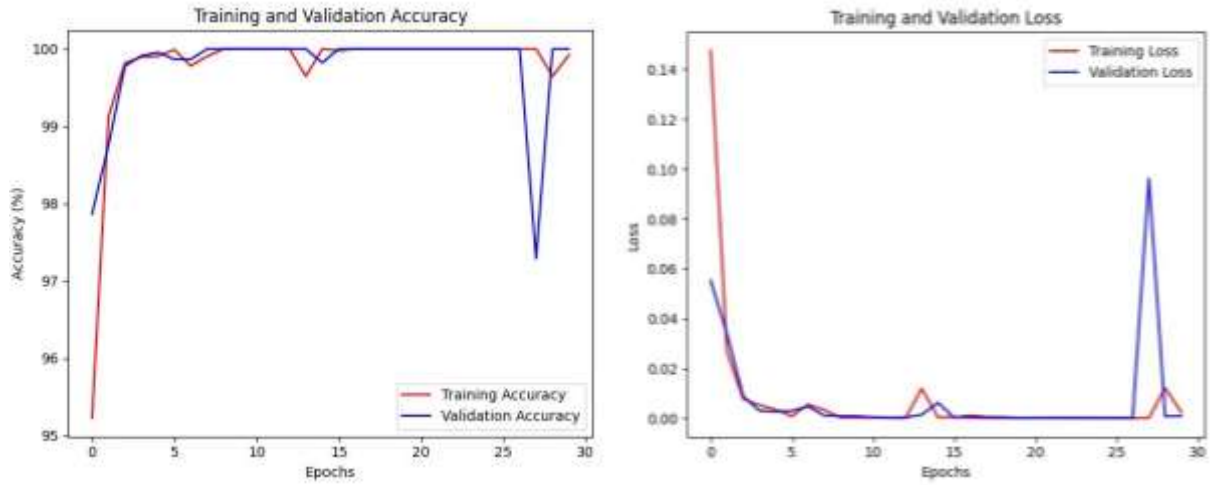


Figure 13. Accuracy and Loss Curves of the DeiT

2) Testing Results Analysis

The categorization performance of the DeiT model for various classes is graphically depicted using a confusion matrix, as illustrated in Figure 14. The results of the confusion matrix show that out of 2,250 test images, 2,249 were correctly predicted, with only one misclassification, demonstrating the exceptional performance of the DeiT model on complex small-sample medical images.

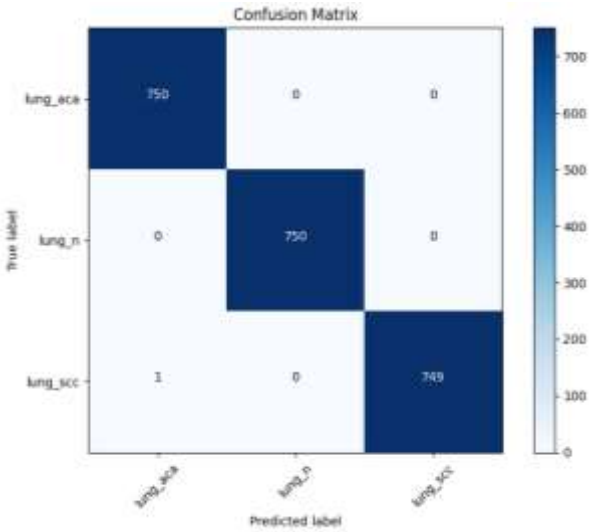


Figure 14. Confusion Matrix

Assess the model with the aid of the test set. The model's performance is assessed by calculating the loss and accuracy on the test set, along with generating a detailed classification report. This report presents the precision, recall, and F1-score for each category. Using these metrics, the overall average precision, recall, F1-score, and macro average are computed. The formulas for each test metric are presented below in Equations (9-13).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

$$Precision = \frac{TP}{TP+FP} \tag{10}$$

$$\mathrm{Re}call = \frac{TP}{TP + FN} \tag{11}$$

$$F1-Score = 2 \times \frac{\mathrm{Pr}ecision \times Recall}{Precision + Recall} \tag{12}$$

$$Macro\ Average = \frac{1}{N}\sum_{i=1}^{N} Metric_i \tag{13}$$

In this context, TP represents true positives, and TN stands for true negatives. The greater the values of TP and TN, the superior the model's predictive capability. FP refers to false positives,

while FN represents false negatives. The smaller the values of FP and FN are, the fewer errors the model commits in its predictions. N denotes the total number of categories. $Precision_i$ signifies the precision for category i, whereas $Recall_i$ denotes the recall for category i, which assesses the proportion of actual positive samples that are correctly identified. A higher recall means the model identifies more positive cases, reducing the likelihood of missing them. $F1\text{-}Score_i$ denotes the F1-score for class i, which is the harmonic mean of precision and recall, effectively balancing both metrics. A higher F1-score indicates better overall predictive performance. $Metric_i$ refers to the precision, recall, or F1-score of class i, while W_i represents the number of samples in class i. Table 2 below presents the evaluation metrics calculated after the model's execution.

TABLE II. PREDICTION METRICS FOR DIFFERENT MODELS

Model	Acc(%)	Average Precision(%)	Average Recall(%)	Average F1-Score(%)
Vgg16	98.49	98.49	98.49	98.49
Resnet50	97.51	97.51	97.51	97.51
DeiT	99.96	99.96	99.96	99.96

The test results indicate that the DeiT model performs the best. Leveraging self-attention mechanisms, it effectively captures long-range dependencies in images and employs knowledge distillation techniques, making it particularly suitable for complex medical image classification tasks on small datasets. To provide a more intuitive demonstration of the model's performance, several samples were randomly selected from the test set, comparing the DeiT model's predictions with the ground truth labels[16]. As shown in Figure 15, these images illustrate the model's classification performance across different categories, further validating its exceptional performance on small-sample, complex medical data.

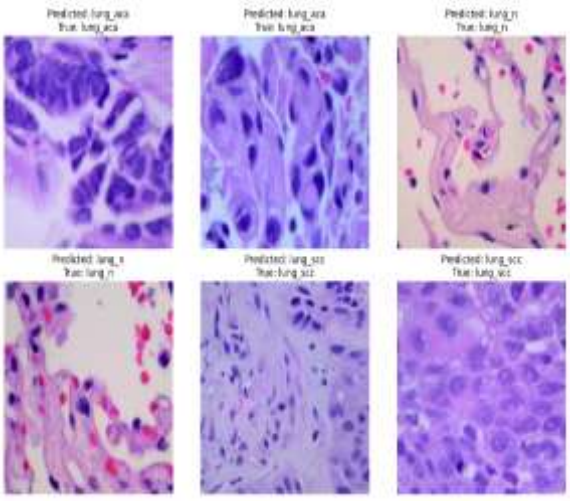


Figure 15. Random Test Plot

V. CONCLUSION AND OUTLOOK

This study explores the task of lung pathology image detection by conducting comparative experiments based on VGG16, ResNet50, and DeiT models. Given the limited dataset size, the experimental results demonstrate that the DeiT model, supported by the self-attention mechanism, can more effectively capture subtle features in pathological images. Additionally, by incorporating the knowledge distillation strategy, DeiT significantly enhances small-sample learning performance. Compared to traditional convolutional neural networks, the DeiT model achieved the highest classification accuracy on the test set, fully showcasing its potential in complex medical image analysis.

Looking ahead, the advantages of the DeiT model lay a solid foundation for novel opportunities in the domain of medical image processing. In broader medical application scenarios, such as the diagnosis of rare diseases, the combination of the DeiT model with knowledge distillation is expected to further demonstrate its capabilities on small-scale datasets, providing strong technical support for the early diagnosis of rare diseases.

REFERENCES

[1] Wu Hongjie, Tian Chuangchuang, Tao Ran, et al. Research on Building Displacement Prediction Method Based on Graph Convolution Distillation

Transformer. Journal of Suzhou University of Science and Technology (Natural Science Edition), 2024, 41(04): 128-138.

[2] Yao Yiying, Chen Junji, Ren Denghong, et al. Case Analysis of Medical Image Recognition and Diagnosis Based on Deep Learning. Application of Integrated Circuits, 2024, 41(12): 80-81.

[3] Liu Yuxin, Meng Yu, Deng Yupeng, et al. A Dual-Stream Extraction Model for High-Resolution Remote Sensing Building Images Integrating CNN and Transformer. Journal of Remote Sensing, 2024, 28(11): 2943-2953.

[4] Li Yunfei, Li Shuting, Zhang Shuai, et al. Research Progress on Deep Learning in Tumor Image Classification. Chinese Journal of Cancer Prevention and Treatment, 2024, 31(12): 719-724.

[5] Zong Haoyu, Qin Yuliang, You Ziyuan. Advances in Deep Learning Applications in Musculoskeletal Imaging. Imaging Research and Medical Applications, 2024, 8(10): 1-3.

[6] Liu Libing, Fu Liyao. Applications and Prospects of Deep Learning Technology in Medical Image Analysis. New Generation Information Technology, 2024, 7(01): 24-28.

[7] Hu Kun, Wu Guoqing, Hu Zuhui, et al. Research on Metal Surface Defect Image Classification Based on an Improved VGG16 Network. Computer Applications and Software, 2024, 41(06): 175-180.

[8] Liu Yansheng, Yu Qianru, Zhang Kun, et al. Establishment and Clinical Testing of a ResNet-Based Model for Colonoscopy Image Classification of Ulcerative Colitis. World Science and Technology - Modernization of Traditional Chinese Medicine, 2024, 26(09): 2346-2354.

[9] Lin Hailin, Chen Guoming, Tang Peiyu, et al. A Lightweight Image Classification Method Based on Convolutional Vision Transformer Fusion. Modern Computer, 2024, 30(22): 1-7.

[10] Chen Ning, Liu Fan, Dong Chenwei, et al. Few-Shot Image Classification Based on Local Contrastive Learning and New Class Feature Generation. Pattern Recognition and Artificial Intelligence, 2024, 37(10): 936-946.

[11] Wang Haibao, Liu Hongyan, Wei Zhi, et al. Research on Bone Marrow Cell Image Classification Based on Deep Learning. Genomics and Applied Biology, 2024, 43(Z2): 1872-1882.

[12] Gong Xuanjin. Long-Tailed Visual Recognition Method Based on Multi-Classifier Hierarchical Distillation. Modern Information Technology, 2024, 8(16): 49-52+59.

[13] Zhao Hongwei, Wu Hong, Mark, et al. An Image Classification Framework Based on Knowledge Distillation. Journal of Jilin University (Engineering Edition), 2024, 54(08): 2307-2312.

[14] Zhou Chengyang, Liu Wei, Wu Tianrun, et al. Rock Thin Section Image Classification Based on a Hybrid Expert Model. Journal of Jilin University (Science Edition), 2024, 62(04): 905-914.

[15] Zhang Li, Yang Minghui, Sun Jiacheng. Few-Shot Tea Leaf Disease Recognition Based on Attention Mechanism and Transfer Learning. Journal of Chinese Agricultural Mechanization, 2024, 45(10): 262-268.

[16] Zhao Tingting, Gao Huan, Chang Yuguang, et al. Fine-Grained Image Classification Method Based on Knowledge Distillation and Target Region Selection. Computer Applications Research, 2023, 40(09): 2863-2868.

.

Research on the Improvement of Image Super Resolution Reconstruction Algorithm Based on AWSRN Model

Bin Dong
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: 2670538455@qq.com

Zhiyi Hu
Engineering Design Institute
Army Research Laboratory
Beijing, 100042, China
E-mail: 763757335 @qq.com

Jun Yu
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: yujun@xatu.edu.cn

Feng Xiong
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, Shaanxi, China
E-mail: 1023465977@qq.com

***Abstract*—In domains such as medical diagnostics, surveillance technology, and geospatial imaging, the escalating need for ultra-high-definition imagery has exposed the limitations of conventional super-resolution methods. These legacy algorithms often fail to deliver the precision and clarity demanded by modern applications. Therefore, this article proposes an optimization algorithm based on the AWSRN network model, aiming to achieve efficient image super-resolution reconstruction, reduce computational costs, and enhance image realism. Firstly, optimize the internal structure of the network and enhance its feature extraction and fusion capabilities; Secondly, to enhance feature extraction precision, a novel module integrating depth-separable convolution with an attention-based mechanism is proposed. Additionally, a hybrid loss function- merging perceptual quality metrics with adversarial training objectives-is employed to rigorously evaluate the disparity between generated and ground-truth images. The MPTS training strategy further optimizes convergence efficiency. Empirical evaluations demonstrate that the enhanced AWSRN model achieves substantial improvements over its baseline counterpart across multiple upscaling factors, particularly at 4× magnification. Specifically, on the Urban100 benchmark, the proposed method elevates PSNR by 1.06 points and SSIM by 0.0239, while maintaining computational efficiency. These advancements offer valuable insights for high-fidelity image upscaling methodologies.**

Keywords-Deep Learning; Image Super-Resolution Reconstruction; AWSRN Network Architecture; Algorithm Optimization

I. INTRODUCTION

In recent years, image super-resolution enhancement has emerged as a prominent focus in visual data optimization. Its primary objective is to reconstruct high-definition visuals from their degraded low-quality counterparts, addressing the growing need for enhanced image fidelity. This approach demonstrates significant applicability across diverse domains, including medical diagnostics, surveillance systems, and satellite imagery analysis. However, super-resolution reconstruction technology also has its shortcomings. Firstly, the super-resolution reconstruction algorithm is computationally complex and requires strict hardware computing resources, which limits its application in scenarios with high real-time requirements or hardware limitations. Secondly, in terms of texture detail restoration, reconstructed images are prone to issues such as artifacts and blurriness, which may result in discrepancies with high-resolution real images. Thirdly, existing models have poor generalization ability, and models trained on specific datasets have poor reconstruction performance when faced with complex and diverse real-world images.

The field of computer vision has witnessed remarkable advancements in enhancing image

DOI: 10.2478/ijanmc-2025-0015

resolution through deep learning techniques in the past decade. Initial approaches relying on interpolation-based algorithms and conventional modeling techniques demonstrated constrained capabilities, until neural network methodologies revolutionized this domain. A pivotal development occurred when researchers led by Chao Dong introduced SRCNN (Super-Resolution Convolutional Neural Network), marking the inaugural successful implementation of CNN architectures for resolution enhancement tasks. This framework employed direct training to establish nonlinear transformations between degraded and high-quality image spaces, yielding substantially superior results compared to classical approaches [1].

Subsequent architectural innovations continued to push performance boundaries. The VDSR (Very Deep Super-Resolution) architecture, developed in 2016, enhanced output quality through increased network depth, demonstrating measurable improvements in quantitative metrics including peak signal-to-noise ratio, thereby producing outputs with greater fidelity to reference high-definition images [2]. That same year saw the introduction of DRCN (Deeply-Recursive Convolutional Network), which implemented parameter-sharing through recursive connections, achieving comparable accuracy with reduced computational overhead and more efficient resource utilization [3].

Building upon these foundations, subsequent architectural refinements yielded continuous improvements. The 2016-introduced VDSR architecture enhanced reconstruction fidelity through network depth expansion, demonstrating superior performance on quantitative assessment measures including signal-to-noise ratio metrics, thereby producing outputs with enhanced objective quality relative to reference high-definition images [4]. That same year saw the development of DRCN, which employed recursive connectivity patterns to achieve parameter efficiency without compromising reconstruction precision, thereby optimizing computational resource utilization [4].

A significant advancement emerged in 2019 with Chaofeng Wang's team introducing AWSRN, an adaptive learning framework for resolution enhancement. This lightweight architecture incorporated dynamic feature weighting mechanisms that automatically adjusted fusion parameters based on regional importance within the input image, substantially enhancing both output realism and processing efficiency [5]. However, the model exhibits limitations when processing complex scenes or non-standard textures. Particularly for artistic imagery with distinctive color distributions and textural patterns - characteristics often divergent from conventional training datasets - the system frequently generates artifacts including distorted textures and inaccurate color reproduction, ultimately failing to preserve the original stylistic integrity in the enhanced outputs [6].To address these challenges, our study presents architectural refinements to both the adaptive weighting residual components (AWRU) and region-specific feature integration modules (LRFU). These modifications strengthen the network's capacity for hierarchical feature processing within localized receptive fields (LFBs), while an enhanced multi-scale weighting mechanism (AWMS) improves handling of intricate textural patterns. The framework further incorporates: (1) a hybrid optimization objective combining multiple loss terms, and (2) an adaptive training protocol, collectively enhancing reconstruction fidelity. Experimental validation demonstrates consistent superiority over baseline AWSRN across standard benchmarks (B100/Urban100), with measurable gains in both PSNR and structural similarity metrics, confirming the efficacy of our design improvements.

II. Image Super-Resolution Reconstruction Model Based On Awsrn

The core competitiveness of AWSRN lies in its unique network architecture and module design, which efficiently achieves image super-resolution reconstruction. The AWSRN network architecture diagram is shown in Figure 1, which includes two core modules: Local Fusion Block (LFB) and Adaptive Weighted Multi Scale (AWMS) Module.

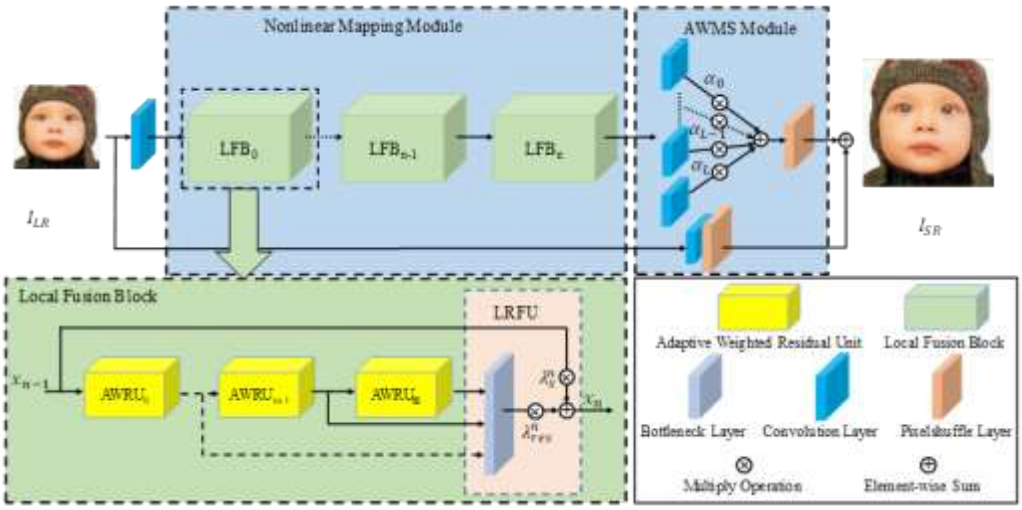


Figure 1. AWSRN Network Architecture Diagram

The LFB module is ingeniously integrated from stacked Adaptive Weighted Residual Units (AWRU) and Local Residual Fusion Units (LRFU). The AWRU unit introduces an adaptive weighting mechanism that can dynamically adjust weights based on the importance of different features, directing the model's attention toward salient features that substantially enhance reconstruction outcomes. This adaptive weighting strategy enhances the efficiency of information and gradient flow without introducing additional parameters, achieving precise learning of image residual information. The LRFU unit further leverages the advantages of local residual fusion, effectively integrating residual information from different AWRU units and enhancing the network's expressive power.

The AWMS module has the ability to fully explore feature information in the reconstruction layer. This module embeds multiple convolutional branches at different scales, which can capture detailed information at different scales in the image. By utilizing convolution operations at multiple different scales, the AWMS module adaptively adjusts the weights of each branch, effectively reducing redundant calculations while ensuring performance. In addition, the AWMS module will intelligently remove redundant scale branches based on the evaluation of network contributions using adaptive weights, and only retain branches that significantly contribute to the reconstruction results. This adaptive weighted multi-scale structure not only improves the efficiency of the network in utilizing feature information, but also significantly enhances the reconstruction performance.

III. Optimization and Improvement Based On AWSRN

Through in-depth analysis of the network structure, reconstruction method, loss function, and training strategy of AWSRN, we propose a series of innovative improvement measures. These architectural refinements simultaneously augment the network's detail preservation capacity while elevating visual quality metrics including sharpness, structural consistency, and photorealistic fidelity.

A. Improvement of Network Structure

The AWSRN network architecture diagram is shown in Figure 2. Firstly, for the optimization of Local Feature Blocks (LFBs), Our optimization efforts primarily targeted the enhancement of Adaptive Weighted Residual Units (AWRUs) and Local Residual Fusion Units (LRFUs) performance characteristics. At the AWRU level, an innovative global context aware weight learning mechanism has been introduced. Extracting global contextual features through Global Average Pooling (GAP), as shown in formula (1).

$$g = \frac{1}{W \times H} \sum_{i=1}^{H} \sum_{j=1}^{W} X(i, j) \quad (1)$$

In this context, $X \in R^{H \times W \times C}$ represents the input feature map, where H and W denote the spatial dimensions of height and width, while the value C corresponds to the total number of channels. $g \in R^{1 \times 1 \times C}$ is the global contextual feature vector. This mechanism not only considers the importance of local features, but also combines global contextual information, by using a two-layer fully connected network (FC) to learn channel attention weights, adaptive weight learning is achieved as shown in formula (2).

$$\alpha = \sigma(W_2 \cdot \delta(W_1 \cdot g + b_1) + b_2) \quad (2)$$

Within this framework, $W_1 \in R^{C/r \times C'}$ and $W_2 \in R^{C' \times C/r}$ denote adjustable weight parameters optimized during training. r represents the compression factor, fixed at 16, while b_1 and b_2 correspond to bias components. The nonlinear activation operator δ is implemented using ReLU, used to normalize weights to the range of [0,1], and $\alpha \in R^{1 \times 1 \times C}$ is the learned channel attention weight. This mechanism can achieve more accurate weight allocation. This improvement significantly enhances AWRU's ability to capture image detail information, providing richer and more accurate feature representations for subsequent image reconstruction.

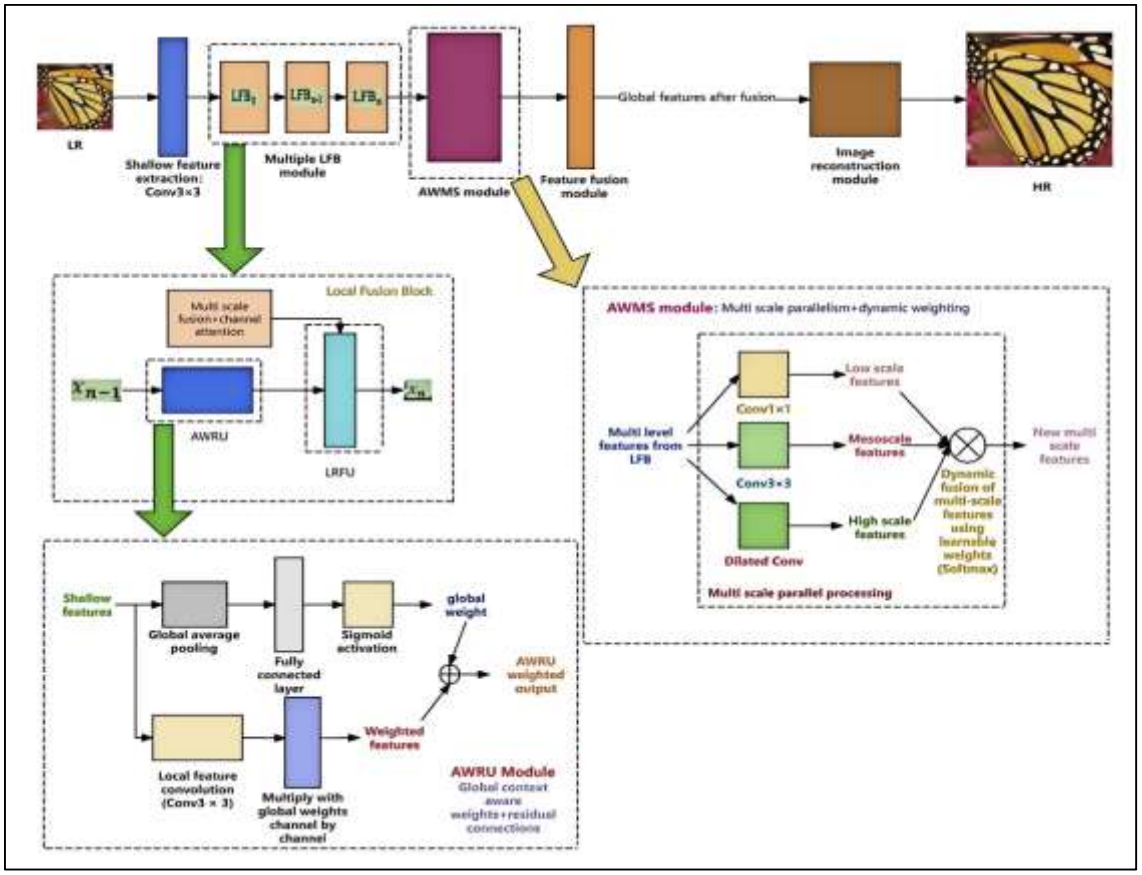


Figure 2. Improved AWSRN network architecture diagram

Secondly, at the LRFU level, we adopted a more optimized feature fusion strategy. Through the incorporation of hierarchical feature integration and focus weighting modules, effective exploitation of multi-level feature representations is accomplished， thereby further improving the quality of reconstructed images. This improvement enables LRFU to more effectively integrate features from different scales and levels, providing more comprehensive and refined feature support for image reconstruction.

In addition, we also optimized the Adaptive Weighted Multiscale (AWMS) module. By introducing richer scale information and more efficient feature extraction strategies, the adaptability of the AWMS module to different scale image features has been significantly improved. This improvement not only enhances the feature extraction capability of the AWMS module, but also increases its sensitivity to image details, thereby further improving the overall performance of AWSRN.

B. Improvement of Reconstruction Methods

Given the challenges posed by diverse image textures, intricate edge details, and noise artifacts, our proposed framework (Figure 3) introduces a novel super-resolution pipeline designed to optimize feature capture completeness and fusion accuracy, thereby advancing reconstruction fidelity.

Our architecture's feature representation phase incorporates a sophisticated unit merging spatially-efficient convolutional decomposition and dynamic feature recalibration. The decomposed convolution process involves: (1) independent spatial filtering per channel, and (2) linear channel combination. The formula for deep convolution is shown in formula (3).

Input feature map $X \in R^{H\times W\times C}$,deep convolution kernel $K \in R^{K\times K\times C}$, output feature map $Y_{depth} \in R^{H\times W\times C}$.

$$Y_{depth}(i,j,c)=\sum_{m=0}^{k-1}\sum_{n=0}^{k-1}K(m,n,c)\cdot X(i+m,j+n,c) \quad (3)$$

Among them, (i, j) is the spatial position, and c is the channel index. The pointwise convolution is shown in formula (4).

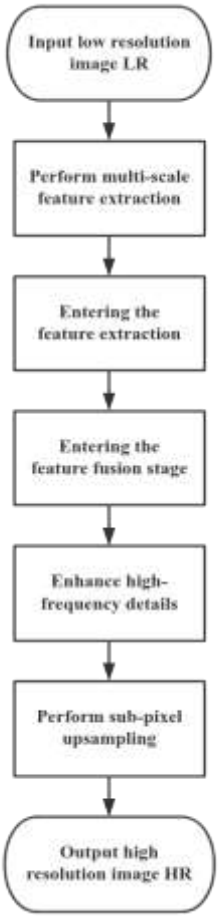


Figure 3. Innovative image super-resolution reconstruction process

Among them, (i, j) is the spatial position, and c is the channel index. The pointwise convolution is shown in formula (4).

Point by point convolution kernel $W \in R^{1\times 1\times C\times C'}$, output feature map $Y_{point} \in R^{H\times W\times C'}$.

$$Y_{point}(i,j,c')=\sum_{C=0}^{C-1}W(1,1,c,c')\cdot Y_{depth}(i,j,c) \quad (4)$$

Among them, c' is the output channel index.

This module not only efficiently extracts multi-scale features from input low resolution images, but also adaptively weights the feature maps

through attention mechanisms. The attention module computes importance scores $A \in R^{H \times W \times C'}$ to dynamically adjust feature representations. These attention coefficients are derived using Equation (5).

Input feature map Y_{point} and generate attention weight A through fully connected or convolutional layers.

$$A = \sigma(W_a \cdot Y_{point} + b_a) \quad (5)$$

Among them, W_a and b_a are learnable parameters, where σ represents the nonlinear activation operation.

The weighted combination of feature representations follows the derivation in formula (6).

Weighted feature map $Y_{att} \in R^{H \times W \times C'}$.

$$Y_{att}(i, j, c') = A(i, j, c') \cdot Y_{point}(i, j, c') \quad (6)$$

By introducing this module, the sensitivity of the model to key image information has been enhanced. The integration of deep features with attention-based weighting enhances the effectiveness of feature learning, but also ensures the richness and accuracy of feature representation.

In the feature fusion stage, we designed a refined fusion strategy based on Long-range structural recurrence and channel dependence. Long-range structural recurrence is used to capture long-range dependencies between feature maps. This strategy calculates the similarity between feature maps, and the calculation process is as follows:

For an input feature representation $X \in R^{C \times H \times W}$ (channel depth C, spatial size $H \times W$), the long-range dependency operations are formulated in (7).

$$Y_i = \frac{1}{C(X)} \sum_{\forall j} f(X_i, X_j) g(X_j) \quad (7)$$

Among them, Y_i is the i-th position of the output feature map. $f(X_i, X_j)$ is a similarity function as shown in formula (8), usually using Gaussian function or dot product to calculate the similarity between positions i and j.

$$f(X_i, X_j) = e^{\theta(X_i)^T \phi(X_j)} \quad (8)$$

Among them, θ and ϕ are linear transformations, usually implemented through 1x1 convolution. $g(Xj)$ is a characteristic transformation function as shown in formula (9), usually also a linear transformation.

$$g(X_j) = W_g X_j \quad (9)$$

$C(X)$ is the normalization factor as shown in formula (10).

$$C(X) = \sum_{\forall j} f(X_i, X_j) \quad (10)$$

Channel dependence is used to dynamically adjust the weights of feature maps to enhance the ability to capture high-frequency details. Given the feature map $Y \in R^{C \times H \times W}$, channel dependence can be achieved through the following steps.

Firstly, calculate the channel attention weight $A \in R^{C \times 1 \times 1}$ as shown in formula (11).

$$A = \sigma(W_2 \delta(W_1 GAP(Y))) \quad (11)$$

Among them, $GAP(Y)$ is a global average pooling operation that compresses the spatial dimension of the features to 1×1 spatial resolution, The learnable parameters W_1 and W_2 correspond to the linear transformation weights, while δ denotes the element-wise activation operator. The sigmoid function σ ensures attention coefficients fall within the unit interval. These weights are then used to recalibrate channel features as mathematically defined in (12).

$$Z = A \otimes Y \quad (12)$$

Where $\otimes$ represents channel wise multiplication operation.

The proposed component facilitates cross-spatial feature synthesis, establishing robust connections between distant image regions. Through channel-wise importance modulation, the framework demonstrates enhanced sensitivity to texture details with improved noise suppression. These refinements yield reconstructions with superior definition and more natural visual continuity.

To rigorously validate our approach, we performed extensive training iterations on the DIV2K benchmark, adhering to established super-resolution evaluation protocols. Quantitative comparisons with current AWSRN architectures reveal that our novel feature integration framework delivers superior perceptual quality. The method exhibits particular advantages in processing challenging visual patterns containing intricate structures and sharp transitions.

C. Improvement of Loss Function

The conventional loss formulation adopted from AWSRN studies incorporates both MSE and PSNR-optimized components. While this framework provides basic pixel-level fidelity measurement between reconstructed and reference images, it demonstrates notable limitations in preserving fine structural details, textural patterns, and perceptual authenticity. These traditional loss functions often result in reconstructed images being too smooth, lacking realistic texture details and sharp edges.

To address these limitations, we enhance the AWSRN optimization framework through a hybrid loss formulation integrating perceptual and adversarial components. The perceptual term quantifies feature-level discrepancies in texture, edge, and structural patterns by evaluating VGG-encoded representations (using a pre-trained VGG network in our implementation). Simultaneously, the adversarial component employs GAN-based discriminative evaluation to improve visual authenticity and realism in reconstructions [7].

The perceptual discrepancy metric computes either L1 or L2 norms between the feature activations of reconstructed and reference images, extracted from designated VGG network layers. This formulation, mathematically expressed in Equation (13), serves to reduce semantic-level feature distortions in the output.

$$L\_perceptual=\sum\_l(1/N\_l)*\square\varphi\_l(I\_generated)-\varphi\_l(I\_target)\square\hat{}2 \quad (13)$$

Among them, $L\_perceptual$ represents the perceptual loss, φ_l represents the features extracted by the pre trained network at the l-th layer, $I2$ generated represents the generated image (i.e. reconstructed image), $I2$ target represents the target image (i.e. original image), N1 represents the dimension of the l-th layer features, $\square\hat{}2$ represents the L2 norm (i.e. the square of Euclidean distance).

The computation involves aggregating L2-norm distances across all feature map layers to derive the cumulative perceptual discrepancy metric. This quantitative measure evaluates the divergence between reconstructed and reference images within deep feature representations.

The adversarial optimization framework employs a discriminative network trained to differentiate super-resolved outputs from ground truth samples, while simultaneously optimizing the generator (AWSRN architecture) to produce visually plausible results capable of bypassing this discrimination, as mathematically formulated in Eq. (14).

$$L\_adversarial=-E[\log(D(G(z)))] \quad (14)$$

Among them, $L\_adversarial$ represents adversarial loss, D represents discriminator, G represents generator (i.e. AWSRN), z represents random noise or low resolution image input to the generator, $E[...]$ represents expectation.

The proposed optimization framework combines perceptual and adversarial losses through weighted summation, yielding the final objective function as defined in Equation (15).

$$L\_total=\lambda\_perceptual*L\_perceptual+\lambda\_adversarial*L\_adversarial \quad (15)$$

Among them, L_total represents the total loss function, while $\lambda\_perceptual$ and $\lambda\_adversarial$ represent the weight coefficients of perceptual loss and adversarial loss, respectively.

D. Training Strategy Improvement

To address AWSRN's training challenges including slow convergence, local optimum trapping, and inadequate high-frequency feature learning, we develop a Multi-Phase Training Scheme (MPTS). This framework implements: (1) A hierarchical curriculum learning approach that incrementally processes images from reduced to full resolution, enhancing detail learning [8]; (2) An adaptive learning rate mechanism incorporating cosine annealing with warm restarts for optimized convergence; (3) A Feature-Adaptive Sample Selection (FASS) module that prioritizes high-information-content samples based on feature distribution analysis [9].

IV. Experiment and Analysis

A. Train Data Set

The DIV2K benchmark comprises 800 training and 100 validation images in high-resolution (HR) format, all exhibiting exceptional visual quality with well-preserved fine details. This collection serves as an excellent resource for developing and testing super-resolution methods. A key advantage of this dataset is its systematic degradation pipeline, which allows generation of low-resolution (LR) counterparts at multiple magnification levels (×2, ×3, ×4) through automated scripts. This standardized preprocessing ensures dataset uniformity while simplifying experimental setup [10].

Furthermore, DIV2K incorporates both bicubic interpolation and configurable degradation models to generate more authentic low-resolution counterparts, enabling comprehensive evaluation of SR methods. The dataset's premium-quality images serve as an optimal training basis for AWSRN, with diverse samples enhancing the network's adaptability and reconstruction quality. The validation subset facilitates rigorous assessment of model precision and stability, verifying practical deployment readiness.

The DIV2K dataset's superior image quality establishes an optimal training basis for AWSRN, with its diverse samples significantly enhancing the network's cross-domain adaptability and reconstruction fidelity. For performance verification, the dedicated validation set enables comprehensive assessment of the model's precision and stability, confirming its operational effectiveness in real-world scenarios [11].

B. Experimental Hardware Configuration

The hardware configuration of this experiment adopts AMAX workstation, equipped with Intel Xeon Gold 6254 processor (18 cores, 36 threads, main frequency 3.1GHz) and 32GB DDR4 2666MHz ECC memory, ensuring high-performance computing and data reliability. The operating system is Ubuntu 18.04 LTS, and the graphics card is NVIDIA GeForce RTX 2080 Ti (11GB GDDR6 VRAM), supporting CUDA and cuDNN acceleration, suitable for training deep learning models. The storage is configured as a 1TB NVMe SSD for fast reading and writing of datasets and model files.

C. Experimental Process

To validate our optimization approach for super-resolution generation, we first trained the model using DIV2K data. For quantitative evaluation, benchmark datasets B100 and Urban100 were employed, with reconstruction quality assessed through two established metrics: PSNR (quantifying pixel-level accuracy) and SSIM (measuring structural preservation). These measurements enable systematic comparison between generated and reference high-resolution images, providing objective performance evaluation. The experimental procedure consists of:

a). Data curation involves systematically pairing high-resolution source images with their synthetically degraded counterparts (generated through resolution reduction) to create organized training and evaluation subsets.

b). The experimental framework involves implementing an adaptive-weighted SR network in Python, incorporating structural refinements to the Local Fusion Block (LFB). Key enhancements include optimizing the Adaptive Weighted

Residual Unit (AWRU) and Local Residual Fusion Unit (LRFU), along with improvements to the Adaptive Weighted Multi-Scale (AWMS) module, all trained using the Adam optimizer.

c). Data preprocessing: Preprocessing the selected image data, including image normalization, cropping, scaling, and other operations, in order to input it into the model for training and testing.

d). The training phase involves feeding low-resolution (LR) samples from the benchmark dataset into the network, with corresponding high-resolution (HR) images serving as ground truth targets for supervised learning.

e). During the assessment phase, the optimized network processes low-resolution test samples to perform super-resolution restoration. Comparative visual results in Figure 4 demonstrate enhanced detail preservation, with panel (a) displaying baseline AWSRN outputs and panel (b) showing our improved reconstruction quality.

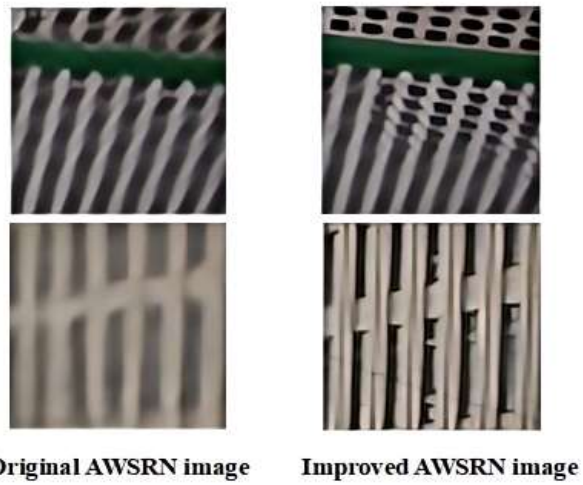


Figure 4. Comparison of image reconstruction details

To quantitatively evaluate reconstruction quality, we employ two established metrics: PSNR measures pixel-level fidelity, while SSIM assesses structural preservation relative to ground truth HR references. Quantitative comparisons at ×4 magnification appear in Table I, with corresponding ×8 results presented in Table II.

TABLE I. QUANTITATIVE COMPARISON ON A SCALE FACTOR OF 4

Scale	Model	B100 PSNR/SSIM	Urban100 PSNR/SSIM
4	AWSRN	27.64/0.7385	26.29/0.7930
	Improved AWSRN	28.47/0.7592	27.35/0.8169

TABLE II. QUANTITATIVE COMPARISON ON A SCALE FACTOR OF 8

Scale	Model	B100 PSNR/SSIM	Urban100 PSNR/SSIM
8	AWSRN	24.80/0.5967	22.45/0.6174
	Improved AWSRN	25.32/0.6214	23.18/0.6438

D. Experimental Results

Quantitative analysis reveals consistent performance gains across all test conditions. For 4× super-resolution, the enhanced ASWRN architecture demonstrates measurable improvements over baseline AWSRN, with B100 dataset showing PSNR/SSIM gains of +0.83 dB/+0.0207 and Urban100 achieving +1.06 dB/+0.0239 improvements. At 8× magnification, quality metrics further improve, with B100 registering +0.52 dB/+0.0247 and Urban100 showing +0.73 dB/+0.0264 enhancements. These progressive gains with increasing scale factors confirm the optimization's effectiveness in bridging the gap between reconstructed and reference images.

The empirical analysis confirms the enhanced AWSRN framework's efficacy in single-image super-resolution applications, demonstrating substantial improvements in both computational efficiency and reconstruction fidelity compared to existing approaches. This optimized architecture achieves superior high-frequency detail recovery and perceptual quality, offering valuable insights for advancing SR algorithm development and computer vision applications [12].

V. CONCLUSIONS

This study proposes an improved AWSRN super-resolution network algorithm that effectively addresses the efficiency and optimization issues in

image super-resolution reconstruction. By optimizing the LFB, LRFU, and AWMS modules, the ability to capture details and reconstruction results have been significantly improved. Empirical evaluations on B100 and Urban100 benchmarks demonstrate consistent metric improvements (PSNR/SSIM) in the enhanced model, with reconstructions exhibiting superior fidelity to ground truth references. This framework introduces novel paradigms for single-image super-resolution while advancing computer vision methodologies. Future directions include: (1) architectural refinements for scenario-specific performance tuning, (2) development of robust optimization protocols to enhance model stability and computational efficiency. The current module-level optimizations establish a strong foundation for subsequent algorithmic developments.

REFERENCES

[1] Dong, C., Loy, C. C., & Tang, X, ″Accelerating the Super-Resolution Convolutional Neural Network,″ in Proc. European Conference on Computer Vision (ECCV), Springer, pp. 391–407, 2020.

[2] J. Kim, J. K. Lee, and K. M. Lee, ″Deep Recursive Residual Network for Image Super-Resolution,″ in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3566–3575, 2021.

[3] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, ″ Residual Recursive Network for Image Super-Resolution,″ in Proc. AAAI Conference on Artificial Intelligence (AAAI), vol. 36, no. 3, pp. 3456−3464, 2022.

[4] Wang, X., Yu, K., Dong, C., & Loy, C. C. (2021). ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 63-79). Springer.

[5] C. Wang, Z. Li, and J. Shi, ″Lightweight Image Super-Resolution with Adaptive Weighted Learning Network,″ arXiv preprint arXiv:1904.02358, 2019.

[6] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, ″ Residual Feature Aggregation Network for Image Super-Resolution,″ in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2358-2367,2022.

[7] X. Wang, K. Yu, C. Dong, and C. C. Loy, ″Recovering Realistic Texture in Image Super-Resolution by Deep Spatial Feature Transform,″ in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 606–615, 2018.

[8] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, ″Image Super-Resolution Using Very Deep Residual C-hannel Attention Networks,″ IEEE Transactions on Patt ern Analysis and Machine Intelligence, vol. 43, no. 10, pp.3551-3567,2021.

[9] X. Chen, J. Wang, and Y. Guo, ″Dynamic Learning Rate Scheduling for Deep Neural Networks with Cosine Annealing and Warm Restarts, ″ Neural Networks, vol. 145, pp. 1–12, 2022.

[10] Z. Wang, J. Chen, and S. C. H. Hoi, ″Deep Learning for Image Super-Resolution: A Survey,″ IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 43, no. 10, pp. 3365–3387, 2021.

[11] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, “Image Super-Resolution Using Very Deep Residual C hannel Attention Networks,” IEEE Transactions on P-at tern Analysis and Machine Intelligence, vol. 43, no. 10, pp. 3551–3567, 2021.

Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu, “Residual Dense Network for Image Super-Resolution,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 5, pp. 2480–2495, 2022.

Research on Driving Conditions Based on Principal Component and K-means Clustering Optimization

Huifeng Wang
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: 1318057134@qq.com

Jiaxiang Fang
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: fangjiaxiang@st.xatu.edu.cn

Shuping Xu
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: 563937848@qq.com

Feiyan Kou
School of Computer Science & Engineering
Xi'an Technological University
Xi'an, China
E-mail: 13992096752@163.com

Huxiang Yang
Shaanxi Coal Industry Chemical Technology
Research Institute Co., Ltd.
Xi'an, China
E-mail: yanghx@sxccti.com

***Abstract*—In order to overcome the problems of traditional K-means algorithm being sensitive to the initial cluster centers and easily affected by noise points, this study proposes an enhanced K-means hybrid clustering algorithm that integrates improved principal component analysis and density optimization. By combining the distance optimization strategy with the density assessment mechanism, a data density evaluation model based on spatial distribution characteristics was established. The algorithm prioritizes data samples with large spacing in high-density areas as the initial cluster center candidate set. It realizes intelligent filtering of abnormal data points while improving the clustering quality, and selects characteristic parameters with higher principal component contribution rates to reconstruct driving conditions, and finally completes the fuel consumption characteristics verification. Experimental data show that the driving conditions constructed by this method have only a 1.17% statistical difference in the speed-acceleration joint probability distribution, and the relative error mean of key characteristic parameters remains at a low level. The research confirms that the constructed driving conditions are statistically significantly consistent with the actual road operation characteristics and can accurately characterize the essential characteristics of traffic flow in a specific area.**

Keywords-Improved Principal Component Analysis; Improved K-Means Clustering; Distance Optimization; Density Method

I. INTRODUCTION

Vehicle driving conditions, also known as vehicle operating cycles or driving cycles, refer to the mathematical representation of the speed-time curve that characterizes the vehicle's operating state in a specific traffic environment [1]. It provides core data support for fuel efficiency evaluation, emission control technology research and development, and intelligent traffic control, and directly affects the design of new energy vehicles and the accuracy of urban traffic carbon accounting [2]. Zhao Xuan's team [3] proposed a driving mode analysis method based on fuzzy C-means clustering. By integrating the time distribution characteristics of kinematic segments with multi-dimensional parameter correlation analysis, they achieved

intelligent identification and optimized reconstruction of typical driving conditions of urban electric vehicles. The operating condition curves they constructed have significant improvements in typicality indicators compared to traditional methods. Currently, K-means clustering is widely used in driving cycle synthesis. However, K-means clustering often has problems such as large dependence on the initial cluster center, isolated points, and sensitivity to noise data. Ma Fumin et al. [4] innovatively constructed a local density dynamic adaptation measurement model to accurately characterize the spatial distribution characteristics of data objects within a cluster, and based on this, designed a rough K-means clustering algorithm that integrates a local density adaptation mechanism. Yuan Yiming et al. [5] developed an optimized K-means text clustering algorithm based on density peak. This algorithm effectively overcomes the convergence instability problem caused by random initialization of centers in the traditional K-means algorithm by accurately selecting density peak points as the initial clustering centers, and significantly improves the reliability of clustering results. Although the above two methods optimize the initial cluster centers to a certain extent, they do not mention the impact of edge data and isolated points in the data set. Bao Zhiqiang et al. [6] only used an outlier removal algorithm to eliminate isolated points in the data set, but still used traditional K-means clustering to cluster the data set.

Based on the above research conclusions, this paper constructs an improved density-driven K-means clustering algorithm. The core innovation of this algorithm is to introduce density measurement indicators to screen the initial clustering centers, effectively suppress the interference of noise data on the selection of initial centers, and integrate enhanced principal component analysis technology to build a two-stage collaborative optimization framework to achieve intelligent synthesis of driving conditions.

II. Data Preprocessing

The measured data obtained in this study are from a light vehicle road operation scenario in a certain city, with a sampling rate of 1 Hz. The data dimensions include multiple source parameters such as timestamp, global positioning system (GPS) speed measurement value, geographic longitude and latitude coordinates, and instantaneous fuel consumption rate. In the actual data collection process, due to the combined influence of factors such as complex driving environment, electromagnetic signal interference, and urban building occlusion, the original sensor data generally has significant noise pollution, which manifests as multiple problems such as data distortion, abnormal value fluctuations, and signal interference [7]. Therefore, the first step in data processing is to preprocess the original data using wavelet decomposition and reconstruction [8]. The basic idea is to remove the wavelet coefficients corresponding to each frequency band and noise while retaining the wavelet coefficients of the original signal, and then perform wavelet reconstruction on the processed coefficients to obtain a pure signal.

Assume that a noisy signal can be described as:

$$S(x) = f(x) + n_1(x) \times n_2(x) \tag{1}$$

Among them, $S(x)$ is the degraded signal, $f(x)$ is the original signal, $n_1(x)$ is the additive noise, and $n_2(x)$ is the multiplicative noise.

The denoising process based on wavelet decomposition and reconstruction is described as follows:

Step 1: Decompose the noisy signal $f(x)$ into approximate component $c_{j,k}$ and detailed component $d_{j,k}$ by wavelet decomposition.

Step 2: According to the threshold δ_j, use equation (2) to process the detailed component $d_{j,k}$ of the layer j.

$$d_{j,k} = \begin{cases} d_{j,k}, & |d_{j,k}| > \delta_i \\ 0, & |d_{j,k}| \le \delta_i \end{cases} \tag{2}$$

Step 3: Use the reconstruction algorithm to

reconstruct the approximate component $c_{j,k}$ and the detailed component $d_{j,k}$ to obtain the filtered signal.

The original data and the data preprocessed by wavelet decomposition and reconstruction are shown in Fig. 1.

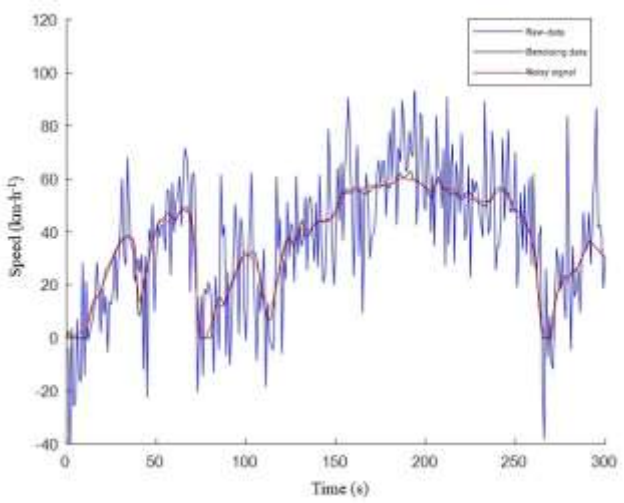


Figure 1. Comparison of noise reduction data results

The comparison between the original data and the preprocessed data shows that the wavelet decomposition and reconstruction method has a good denoising effect and can effectively improve the signal-to-noise ratio of the signal.

III. Analysis of Driving Condition Data

A. Feature Parameter Extraction and Kinematic Segment Division

Based on the analysis of relevant data and related research, 12 characteristic parameters were defined to describe the kinematic segments [8]. This paper selects 12 characteristic parameters including running time T/s, driving distance S/km, average speed $V_a/(km\cdot h^{-1})$, average driving speed $V_d/(km\cdot h^{-1})$, idling time ratio $T_i/\%$, acceleration time ratio $T_a/\%$, deceleration time ratio $T_d/\%$ cruising time ratio $T_c/\%$, speed standard deviation $V_{std}/(km\cdot h^{-1})$, average acceleration $a_a/(m\cdot s^{-2})$, acceleration standard deviation $a_{std}/(km\cdot h^{-1})$, and average deceleration $a_d/(m\cdot s^{-2})$.

The interval from the start of one idle speed to the start of the next idle speed of the car is called a kinematic segment [9]. This paper uses Python language to process and segment 1,655 kinematic segments from 195,815 pre-processed data.

B. Improved principal component analysis

Although the classical principal component analysis can effectively eliminate the differences in dimensions and magnitudes between the original variables when standardizing data, this process may also cause the characteristic differences of different indicators to be over-smoothed, resulting in potential information loss [10]. In view of the above situation, the improved principal component analysis method is as follows:

Step 1. Improve the traditional principal component dimensionless method by using the indicator mean method and indicator homogeneity method [11]. Assume that there are m objects and n indicators in the overall evaluation, and the initial indicators can form a matrix $X_{ij}=(x_{ij})_{m\times m}$. To average the matrix is to divide the original index by the average value of all indexes Y_{ij}:

$$Y_{ij}=x_{ij}/\bar{x}_j,(i=1,2,\cdots,n;\ j=1,2,\cdots,p) \quad (3)$$

Among this:

$$\bar{x}_j=\frac{1}{n}\sum x_{ij},(j=1,2,\cdots,p) \quad (4)$$

The index can be processed to make all indicators have the same effect on the whole in the same direction. In the series, let y_{ij} be the reverse index, $\min_{1\le i\le m}\{y_{ij}\}$ is the smallest number among them, and the index is processed to be:

$$y_{ij}^{'}=y_{ij}-\min_{1\le i\le m}\{y_{ij}\},(i=1,2,\cdots,m;\ j=1,2,\cdots,n) \quad (5)$$

Among them, $y_{ij}^{'}$ is the sequence after y_{ij} is homogenized, Such changes will not change the

distribution of the original indicators. The improved principal component can represent more characteristic parameter information and achieve dimensionality reduction of driving conditions.

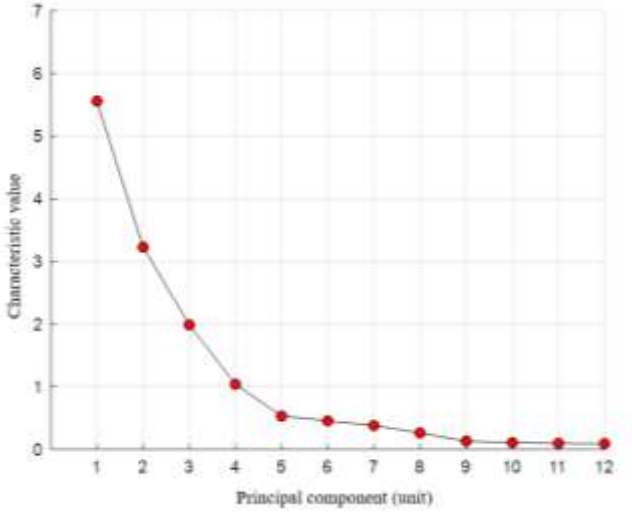


Figure 2. Lithotripsy

Fig. 2 shows that there are obvious inflection points in the variation curves of each principal component, and it is concluded from this observation that the first three principal components are selected.

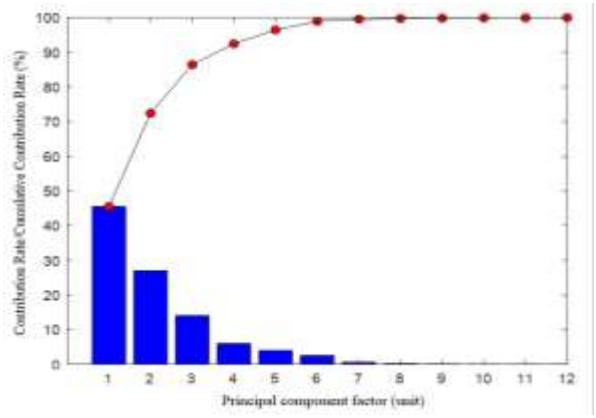


Figure 3. Contribution rate and cumulative contribution rate

As shown in Fig. 3, the cumulative contribution rate of the first three principal components has reached 85%, which basically represents all the information of the 12 characteristic parameters of the fragment, and can be used for cluster analysis. The first principal component contains 45% of the information, thus meeting the requirement of fewer principal components representing more information.

TABLE I. PRINCIPAL COMPONENT LOAD MATRIX

Characteristic parameters	M_1	M_2	M_3
Deceleration time ratio T_d/%	0.323	0.351	-0.223
Driving Clustering S /km	0.893	0.234	0.065
Run time T /s	0.782	0.251	-0.342
Acceleration time ratio T_a/%	0.396	-0.186	0.061
Cruise time ratio T_c/%	0.641	0.335	-0.075
Average speed $V_a/(km \cdot h^{-1})$	0.499	0.763	0.125
Deceleration time ratio $V_d/(km \cdot h^{-1})$	0.778	0.415	0.132
Speed standard deviation $V_{std}/(km \cdot h^{-1})$	0.498	0.333	0.054
Acceleration standard deviation $a_{std}/(km \cdot h^{-1})$	0.125	0.267	-0.077
Average acceleration $a_a/(m \cdot s^{-2})$	0.024	0.523	0.053
Average deceleration $a_d/(m \cdot s^{-2})$	0.266	-0.433	-0.059
Idle time ratio $T_i/(m \cdot s^{-2})$	0.165	-0.351	0.853

The larger the absolute value of the parameter principal component load coefficient, the higher the correlation coefficient between a parameter and a principal component, and the larger the contribution factor. According to Table 1 above, the eigenvalues of the first principal component are driving distance, segment duration, cruising time ratio, and average driving speed; the eigenvalues of the second principal component are average speed; and the eigenvalues of the third principal component are idling time ratio. From the first three principal components, it can be seen that the 12 characteristic parameter matrices of the sample are reduced to 6 characteristic parameter matrices that can represent most of the sample information.

C. Improved K-means cluster analysis

The K-means algorithm is sensitive to the selection of initial cluster centers. Since the process uses a random mechanism, the initial centroids it selects may be distributed in data sparse areas or coincide with outliers. This non-ideal initial state can easily cause the algorithm to fall into a local optimal solution, thereby reducing the clustering quality [12]. In the usual optimization method, in order to make the initial cluster center better than the method of randomly selecting cluster centers in traditional algorithms, k data objects with the farthest distance or the largest density are generally selected as the initial cluster centers. However, if there is noise data in the data set, the "distance optimization method" is likely to use the noise data as the initial cluster center. The "density method" selects the k data objects with the largest density as the initial clustering centers. This method can remove isolated points of data, but it is not suitable

for non-convex data sets. This paper proposes a method that combines the "distance optimization method" and the "density method" to determine the optimal initial clustering center, and constructs a data set density measurement method.

1) Relevant definitions

a) The Euclidean distance between two points in space is:

$$d(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + \cdots + (x_{im} - x_{jm})^2} \quad (6)$$

Among them, x_i , x_j are two m-dimensional data points.

b) Average distance between data objects:

$$MeanDist = \frac{1}{C_n^2} \sum d(x_i, x_j) \quad (7)$$

n is the number of data points in the data cluster, and C_n^2 is the number of logarithms obtained from n data points.

c) Given a data set $D = \{x_1, x_2, \cdots, x_n\}$ *, the density measurement function of data point* x_i *is:*

$$Dean(x_i) = \sum_{j=1}^{n} u(MeanDist - d(x_i, x_j)) \quad (8)$$

Among them, the $u(z)$ function is expressed as:

$$u(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

The density parameter of data point x_i is actually a data object inside a circle with center x_i and radius $MeanDist$.

d) The average density measurement function of the data set is defined as:

$$MeanDens(D) = \frac{1}{n} \sum_{i=1}^{n} Dens(x_i) \quad (9)$$

n is the number of data objects in the dataset a D .

e) For data point x_i *in data set* D *, if*

$$Dens(x_i) < \alpha \times MeanDens(D) \quad (10)$$

Point x_i is called an isolated point, where $0 < \alpha < 1$.

f) The distance between data object x_i *and data set* C *is the closest distance to all data points in data set* C .

$$d(x_i, C) = \min(d(x_i, x_j), x_j \in C) \quad (11)$$

2) Algorithm Idea

The improved K-means clustering process is as follows: first evaluate the density distribution function of all samples, identify and remove outliers, and then construct a high-density data subset. Then select the sample with the best density value as the first initial cluster center, and then select the sample points with the farthest distance from the previous center in the remaining high-density data as new cluster centers, until k initial centroids are established. Finally, the standard K-means clustering process is executed based on the optimized centroid configuration.

The algorithm is described as follows:

Input: Sample dataset $D = \{d_1, d_2, \cdots, d_n\}$ containing n data objects

Output: optimal k value and clustering results.

Step 1: Use $d(x_i, x_j)$ and $MeanDist$ to calculate the distance and average distance between any two data objects in data set D .

Step 2: Use $Dens(x_i)$ and $MeanDens(D)$ to calculate the density measurement function of all data objects in data set D and the average density measurement function of data set D .

Step 3: According to formula (10), determine the isolated data objects and delete them from set D to obtain set A with high density parameters.

Step 4: Select a data object with the highest

parameter density from set A as the first initial cluster center, add it to set B, and remove it from set A.

Step 5: From set A, select the data object farthest from set B as the next initial cluster center, add it to set B, and remove it from set A.

Step 6: Repeat Step 5 until the number of data objects in set B is k.

Step 7 uses traditional K-means for clustering based on k cluster centers.

3) Results Analysis

CH is used as an evaluation index to determine the optimal K value before clustering. It is a measure based on the intra-class dispersion matrix and inter-class dispersion matrix of all samples. The larger the CH is, the tighter the clusters are and the more dispersed the classes are. In this case, the clustering result is relatively better [13]. The index is defined as:

$$CH(k) = \frac{trB(k)/(k-1)}{trW(k)/(n-k)} \tag{12}$$

Where n is the number of clusters, k is the current class, $trB(k)$ is the trace of the between-class dispersion matrix, and $trW(k)$ is the trace of the within-class dispersion matrix.

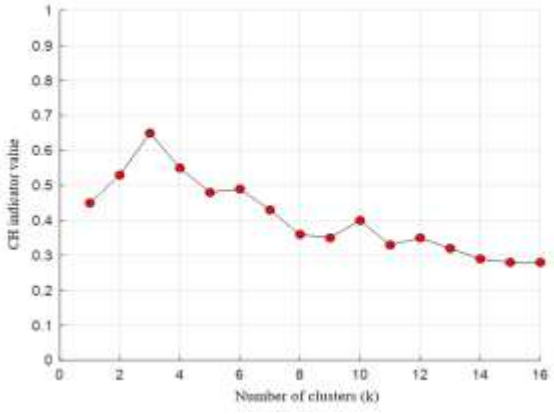


Figure 4. Relationship between cluster number and CH index

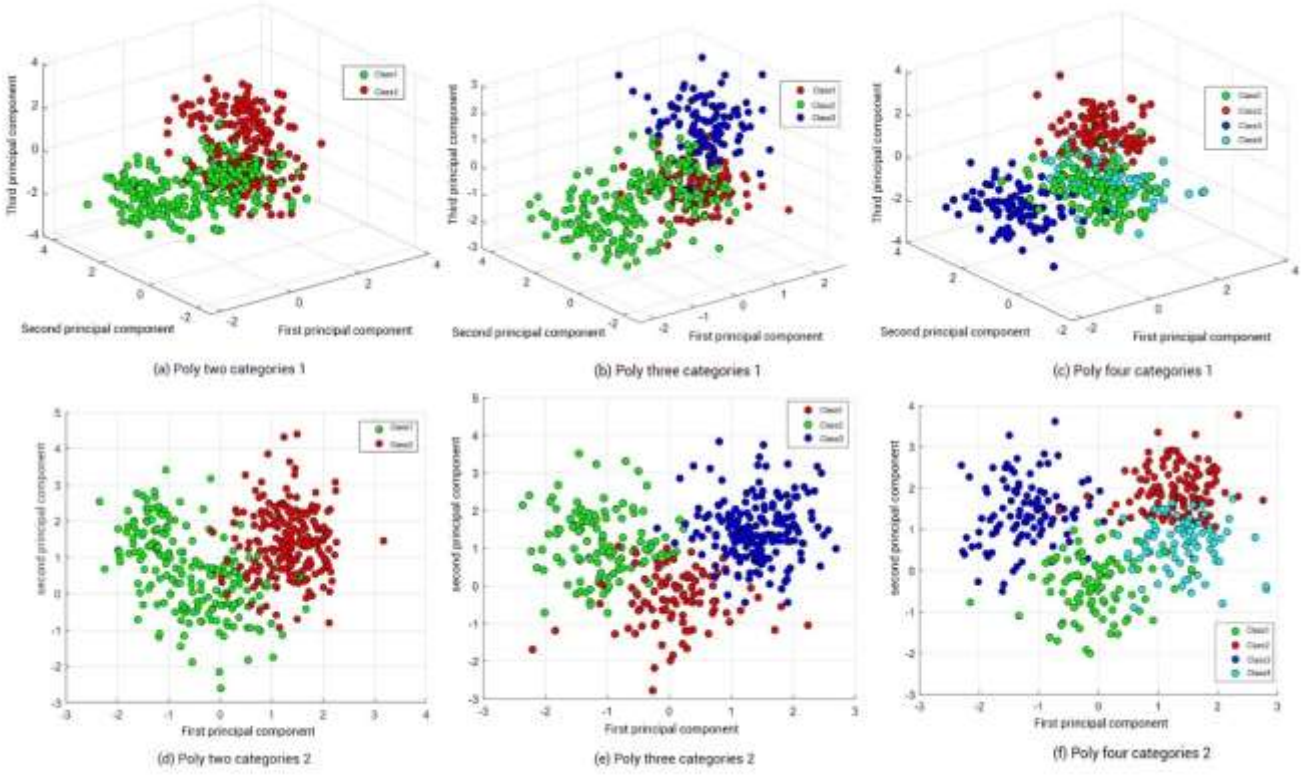


Figure 5. K-means clustering results

In order to determine the appropriate number of clusters, this paper first clusters into 2, 3, and 4 categories, and the clustering results are shown in Fig. 5. At this time, the clustering k value cannot be clearly determined. At this time, CH can be used as an evaluation indicator. The processing results of CH values under different clustering states are shown in Fig. 4. It can be observed that 3 categories are clustered when the CH value is the largest.

Reference [14] used two typical driving condition characteristic parameters, average vehicle speed and idling time ratio, to conduct clustering research. This study innovatively focused on the cruising time ratio and average driving speed, which have a higher contribution, as the core analysis dimensions. The original data points shown in Fig. 6 are scattered point distribution, in which the red marked area clearly circles the isolated samples and outliers that significantly deviate from the main distribution trend.

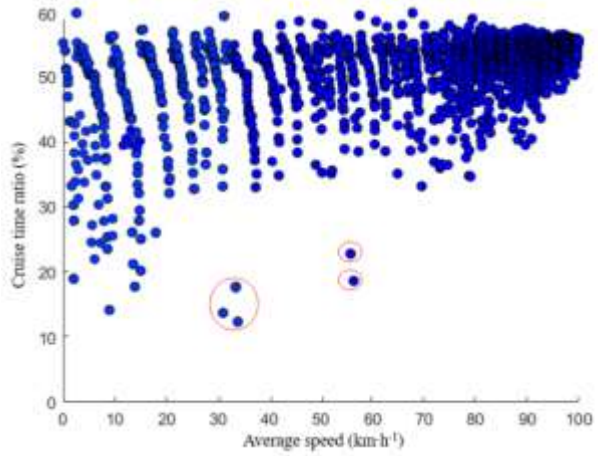


Figure 6. Scatter plot of segments in two-dimensional feature space

This paper clusters the kinematic segments into three categories. As shown in Fig. 7, the cluster centers of the first, second and third categories are (14, 38), (52, 45) and (86, 54) respectively, considering the general urban traffic conditions: the first category is the relatively crowded urban area, with relatively low average speed and cruising time, and more frequent starts and stops; the second category is the relatively unobstructed urban suburbs, with relatively high average speed and cruising time, and fewer starts and stops; the third category is the unobstructed high-speed segment, with high average speed and cruising time, and fewer starts and stops.

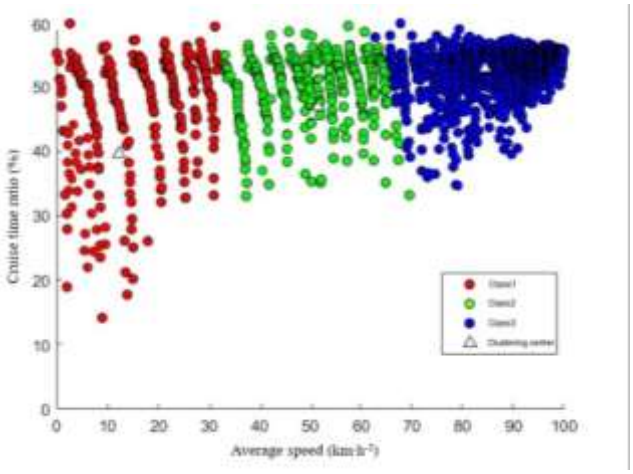


Figure 7. Clustering results of fragments in two-dimensional feature space

IV. Driving Condition Construction and Analysis

A. Working condition construction

According to the proportion of the total time of each time segment to the driving conditions of all data sets, the time taken by each segment in the final constructed condition can be calculated. As shown in Fig. 8, this paper constructs it according to the time of 1,200s of the general typical driving condition.

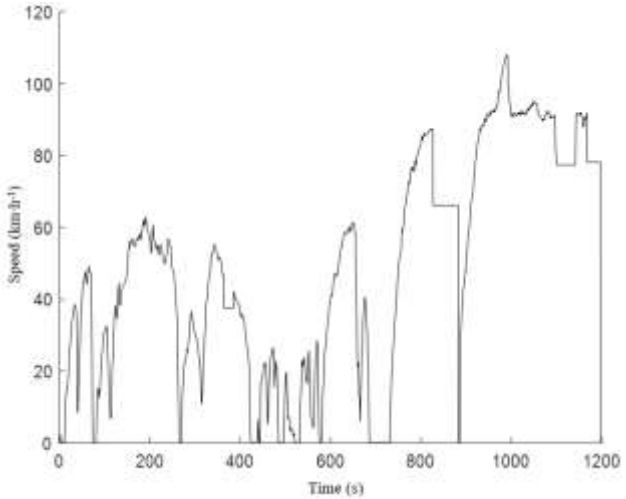


Figure 8. Synthetic driving conditions

As can be seen from Fig. 9, most of the operating points are concentrated in the medium and low speed range, and the distribution of acceleration is relatively reasonable, which can show the actual acceleration and deceleration of the car.

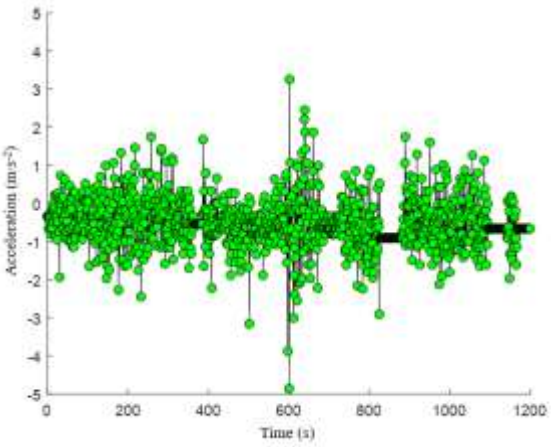


Figure 9. Time and acceleration diagram

As can be seen from Fig. 10, the acceleration is mainly distributed in the speed range of 0-40 $km \cdot h^{-1}$ and around 80 $km \cdot h^{-1}$. During low speed and high acceleration, the instantaneous fuel consumption has a significant bulge, which may be caused by the driver's improper operation.

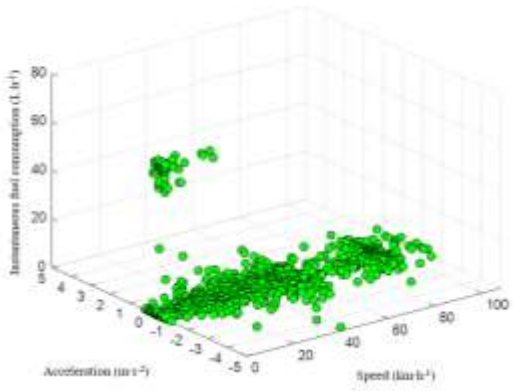


Figure 10. Scatter plot of instantaneous fuel consumption of speed and acceleration

B. Working condition verification and fuel consumption analysis

The smaller the distribution difference value $SAFD_{diff}$ is, the higher the commonality between the constructed working condition and the actual data is [15].

$$SAFD_{diff} = \frac{\sum_i (SAFD_{cycle}(i) - SAFD_{data}(i))^2}{\sum_i SAFD_{data}(i)^2} \tag{13}$$

$SAFD_{cycle}$ is the $SAFD$ of a cycle, and $SAFD_{data}$ is the $SAFD$ of all data.

TABLE II. COMPARISON OF THE METHOD IN THIS PAPER AND TRADITIONAL K-MEANS RESULTS

Method	Eigenvalue mean relative error /%	Cluster average accuracy /%	Average time /s	SAFD^{diff}/%
Traditional K-means clustering	8.1	93	215	2.12
Clustering of this article	4.6	98	127	1.17

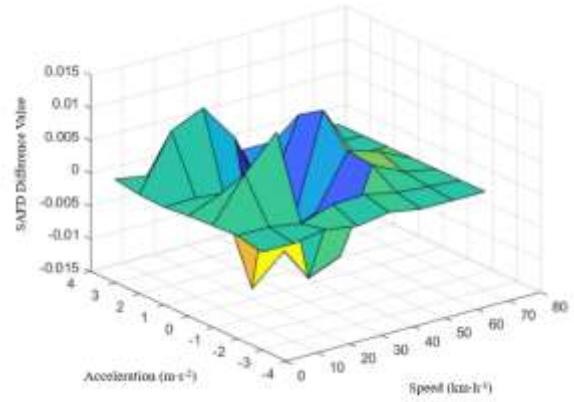


Figure 11. SAFD difference between experimental data and synthetic conditions

The velocity-acceleration joint distribution is to separate the velocity values and acceleration values into equal intervals and further calculate the proportion of the working condition data in different intervals [16]. As shown in Fig. 11, the combined speed-acceleration difference between the original data and the constructed driving condition is distributed within the range of $\pm 1.2\%$, and the calculated distribution difference value ($SAFD_{diff}$) is 1.17%. Therefore, the driving condition constructed in this paper meets the driving characteristics of light vehicles and has strong applicability.

V. CONCLUSIONS

In view of the inherent defects of information distortion in the dimensionless processing of traditional principal component analysis, this study proposes a dual optimization strategy of "mean preservation-trend synchronization" to achieve standardized improvement while maintaining the original difference characteristics of the variables. In view of the limitation of the K-means clustering algorithm being sensitive to the initial center, an intelligent optimization mechanism for initial clustering centers based on local density distribution is constructed. The statistically representative initial centroids are selected by the density measurement value of data objects, which effectively eliminates noise interference and improves clustering stability. This paper constructs an optimization method of improved principal component and improved K-means combination to synthesize automobile driving conditions. The verification results show that the difference rate between the synthetic conditions generated by the proposed method and the original data in the joint distribution space of speed-acceleration is only 1.17%, and the reconstruction accuracy of the conditions reaches 98.83%. The driving conditions synthesized by the proposed method are significantly better than those of traditional methods and are closer to actual traffic conditions.

REFERENCES

[1] YU Yefeng, ZHANG Chen, GAO Zhan, et al. Optimization of Driving Cycle Development Based on Multi-Objective Genetic Algorithm [J]. Chinese Internal Combustion Engine Engineering, 2023, 44(05): 57-65.

[2] HAN Rui, SHI Pengwei, DING Qingguo, et al. Construction of driving conditions for light vehicles on urban roads in Harbin [J]. Technology & Economy in Areas of Communications, 2025, 27(01): 50-58.

[3] Zhao X, Yu Q, Ma J, et al. Development of a Representative EV Urban Driving Cycle Based on ak-Means and SVM Hybrid Clustering Algorithm [J]. Journal of Advanced Transportation, 2018, 2018(1): 1890753.

[4] Ma Fumin, Lu Ruiqiang, Zhang Tengfei. Rough K-means clustering algorithm based on local density adaptive metric [J]. Computer Engineering and Science, 2018, 40(01): 184-190.

[5] Yuan Yiming, Liu Hongzhi, Li Haisheng. Improved K-Means text clustering algorithm based on density peak and its parallelization [J]. Journal of Wuhan University (Science Edition), 2019, 65(05): 457-464.

[6] Bao Zhiqiang, Zhao Yuanyuan, Hu Xiaotian, et al. A new K-Means clustering algorithm that is not sensitive to outliers[J]. Modern Electronic Technology, 2020, 43(05): 109-112.

[7] Ding Yifeng, Li Jun, Gai Hongchao, et al. Application of Wavelet Transform to Vehicle Speed Data Processing for Construction of Driving Conditions [J]. Science Technology and Engineering, 2017, 17(28): 274-279.

[8] Zeng Xiaorong, Kong Lingwen, Yang Xueyi, et al. Application of Principal Component Analysis to Vehicle Driving Conditions [J]. Automobile Practical Technology. 2014(05): 5-9.

[9] Peng Yuhui, Zhuang Yuan. Construction Method of Urban Sanitation Vehicle Driving Conditions Based on Combination Optimized Clustering and Markov Chain[J]. Journal of Fuzhou University (Natural Science Edition). 2019, 47(04): 502-508.

[10] Shang Liqun, Wang Shoupeng. Application of Improved Principal Component Analysis Method in Comprehensive Evaluation of Thermal Power Units [J]. Power System Technology, 2014, 38(07):1928-1933.

[11] Fang Rui. Research on Wuhan City Competitiveness Based on Improved Principal Component Analysis[D]. Huazhong University of Science and Technology, 2012.

[12] Yanling D, Qun L, Shuyin X. An improved initialization center k-means clustering algorithm based on distance and density [C]//American Institute of Physics Conference Series. 2018, 1955(1): 40-46.

[13] Zhang Yuanxiang. Research on the method of determining the optimal number of clusters in cluster analysis [D]. Anhui University, 2020.

[14] Fotouhi A, Montazeri-Gh M. Tehran driving cycle development using the k-means clustering method[J]. Scientia Iranica, 2013, 20(2): 286-293.

[15] Nguyen Y L T, Nghiem T D, Le A T, et al. Development of the typical driving cycle for buses in Hanoi, Vietnam [J]. Journal of the Air & Waste Management Association, 2019, 69(4): 423-437.

[16] Ye Chenchen, Zhang Hongkun, Fan Luyan, et al. Experimental Research on Urban Road Conditions of Passenger Cars in Shenyang City [J]. Science Technology and Engineering, 2017, 17(21): 241-247.

Code Vulnerability Detection Based on Graph Neural Network

Yege Yang
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: 664730726@qq.com

Guiping Li
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: 15693685@qq.com

***Abstract*—Deep learning has emerged as a vital approach for identifying and addressing vulnerabilities in software systems. A key challenge in this process lies in effectively representing code and leveraging AI techniques to capture and interpret its semantics and other intrinsic information. This paper employs bidirectional slicing techniques to extract code slices containing control and data dependencies from program dependency graphs, targeting key points of different vulnerabilities. To represent the node features within the slices, code tokens are mapped to integers and transformed into fixed-length vectors, leveraging Word2vec and BERT models to embed the code nodes and extract structural graph features. The embedded feature matrix is then fed into a Gated Graph Neural Network (GGNN), which aggregates information from nodes and their neighbors to enhance long-term memory of graph-structured data. By iterating through several time steps within GRU units, the final node features are generated. Additionally, edge relationships are used to propagate and aggregate information, further improving the accuracy of vulnerability detection. Experimental results demonstrate that the proposed model achieves an F1-score of 93.25% on the BigVul dataset, showcasing strong detection performance.**

Keywords-Software Security; Deep Learning; Program Analysis; Code Vulnerability Detection; Graph Neural Network

I. INTRODUCTION

A. Background Introduction

Software vulnerabilities are an important cause of network attacks and data leakage, posing a serious threat to software security. Despite efforts to pursue secure programming, vulnerabilities are still widespread due to the increasing complexity of software and the continuous expansion of the Internet. According to the vulnerability data released by China National Vulnerability Database of Information Security (CNNVD) in 2022, there were 24801 new vulnerabilities added in 2022, an increase of 19.28% compared to 2021. The growth rate has markedly increased, reaching record levels and sustaining its upward momentum. The percentage of extremely high-risk vulnerabilities is steadily rising [1]. Vulnerabilities in software, once exploited by malicious attackers, may cause serious consequences such as system paralysis and personal privacy data leakage, posing a ransomware risk to the company or posing a threat to public security. Therefore, ensuring the reliability of software has become a current focus of attention to protect internet users from cyber attackers. Code vulnerabilities often arise from minor errors and can rapidly proliferate due to the extensive use of open-source software and code reuse. Early detection of vulnerabilities to protect software security has become crucial. Detecting and fixing software vulnerabilities is a complex task because of the wide variety of vulnerabilities and their increasing frequency.

In the last dozen years, machine learning(ML) has made significant progress, particularly in areas such as deep learning (DL) algorithms [2], natural language processing techniques [3], and other data-driven approaches, which have proven highly effective in detecting software vulnerabilities. ML/DL models excel at identifying subtle patterns and correlations within large datasets, a critical capability for vulnerability detection, as vulnerabilities often stem from intricate code features and dependencies. These models handle diverse data types and formats, including source code [4-11], textual information [12], and numerical features like submission characteristics [13]. By processing and analyzing these diverse

data representations, ML/DL models enable effective vulnerability detection. This adaptability allows researchers to leverage multiple data modalities for a more comprehensive approach to detecting vulnerabilities.

II. Related Works

DL-based of vulnerability detection methods are currently the forefront of vulnerability detection, which can effectively narrow down the scope of code auditing, avoid expert defined features, and achieve the goal of automatic vulnerability detection [14]. The existing DL-based modeling techniques currently fall into two categories: sequence-based models and graph-based models [32].

A. Sequence Based Model

In the sequence-based modeling approach, the code is considered as a sequence of tokens, which are slices of the code, including methods based on token sequences of function call, source codes, intermediate code, and assembly code, respectively.

Wu et al. [15] employed Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), and CNN-LSTM architectures to model function call sequences from binary programs, generating numerical vector representations of the sequence data through Keras and enhancing vulnerability detection by analyzing function call patterns. Russell et al. [16] reduced token vocabulary size using a custom C/C++ lexer and applied CNN and Random Forest models to detect vulnerabilities at the function level, focusing on the sequential characteristics of source code. Yan et al. [17] introduced the HAN-BSVD model, which captures local sequential features using Bidirectional Gated Recurrent Unit (Bi-GRU) and Text-CNN, with word attention modules highlighting critical sequence regions for binary vulnerability detection. Li et al. [18] developed VulDeeLocator, which utilizes the Bidirectional Recurrent Neural Network (BRNN-vdl) to integrate sequence-based intermediate code analysis, combining data and control flow dependencies to improve vulnerability detection accuracy. Tian et al. [9] proposed BVDetector, utilizing Binary Gated Recurrent Unit (BGRU) and program slicing to analyze sequences of control and data flow, specifically detecting vulnerabilities related to library/API function call sequences in binary programs.

Based on the aforementioned methods, sequence-based vulnerability detection techniques are primarily employed for binary code analysis. These methods effectively capture the program's execution order and logical flow while minimizing the complexity of the analysis. Sequence models demonstrate strong adaptability in processing binary code, particularly in scenarios where source code is unavailable, making them a powerful tool for vulnerability detection.

B. Graph Based Model

The graph-based modeling approach treats the code as a graph and merges different syntactic and semantic dependencies, which can be used for vulnerability prediction using different types of syntactic and semantic graphs in two main ways, transforming the graph structure into a sequence or modeling the graph structure directly.

The Sequence Graph Hybrid Model utilizes both the graph structure and linear sequence in the program. The information extracted from the graph structure is usually transformed into linear sequences, which can be directly input into machine learning models or deep learning models (such as LSTM, GRU) for analysis and prediction. This transformation makes complex graph structure information easier to be processed by traditional sequence models. VulDeePecker [19] extracted code slices from a Data Dependency Graph (DDG) by capturing data flow dependencies between variables and using these slices as input sequences for an RNN and Bi-LSTM model. μVulDeePecker [20] enhanced the detection process by incorporating both data and control dependencies through System Dependency Graphs (SDG), using forward and backward slicing techniques to capture local and global semantic information. These code slices were processed with a Bi-LSTM model to improve vulnerability type identification. SySeVR [21] combined both Bi-LSTM and Bi-GRU models to analyze slices representing data and control flow dependencies, further improving vulnerability

detection by capturing the sequential relationships in these flows. Compared with modeling by converting graph structures into sequences, directly using graph neural networks to model graph structures can more fully learn the dependency relationships between nodes, program structure, and semantic information in graph based code representation. This method can provide more accurate feature vector representations for source code vulnerability detection models. In the method of directly modeling graph structures, one approach is to comprehensively represent the syntax and semantic information related to vulnerabilities at different levels of abstraction by integrating multiple intermediate representations of code, without using slicing techniques. Another approach is to use program slicing techniques to remove information unrelated to vulnerabilities from mixed code representations, in order to improve the accuracy and efficiency of vulnerability detection.

Zhou et al. [22] proposed Devign, which encoded source code into a Combined Program Dependency Graph (CPG), integrating the Abstract Syntax Tree (AST), Control Flow Graph (CFG), and Data Flow Graph (DFG). Natural Code Sequence (NCS) edges were added to preserve code order, and a Gated Graph Recurrent Network (GGRN) with convolutional layers was used for graph-level classification. Wang et al. [23] proposed FUNDED, which improved vulnerability prediction by automatically collecting high-quality samples through confidence prediction (CP). They combined AST and Program Control and Dependency Graphs (PCDG), extracting nine code relationships, and used GGNN with GRU-based models to capture higher-order neighborhood information for detection. Zheng et al. [24] proposed VulSPG, which merged control, data, and function call dependencies into Sliced Program Graphs (SPG) and utilized a Relational Graph Convolutional Network (R-GCN) for vulnerability detection, further enhanced by a triple attention mechanism. Cheng et al. [30] proposed DeepWukong, which generated program slices from Program Dependency Graphs (PDG) and employed GCN and k-dimensional GNN (k-GNN) models to process these slices. Cao et al. [25] introduced MVD, which incorporated function call information into PDGs and focused on detecting memory-related vulnerabilities. The method embedded the program slices using Doc2Vec and employed Flow Sensitive Graph Neural Networks (FS-GNN) to enhance vulnerability detection. Zou et al. [26] utilized PDG-based vulnerability slicing to capture vulnerabilities around pointers and sensitive APIs, employing GGNN models to learn and interpret vulnerability features.

The graph-based modeling approach treats code as graphics and combines different syntax and semantic dependencies. Graph-based representation methods can effectively preserve complex semantic information such as the logical structure and dependency relationships of the code.

III. Feature Extraction

A. Code Standardization

Semantic irrelevant information in source code, such as comments, complex variable names, and function names, can interfere with the training and prediction accuracy of deep learning models. To minimize the impact of this irrelevant information, this study first normalizes the source code. Comments, spaces, tabs, and line breaks are removed while ensuring the corresponding line numbers remain unchanged. Function names are replaced with a standardized identifier (e.g., FUNC1) under specific conditions: they are not keywords (e.g., boolean, const), preprocessor directives (e.g., #define, #include), library functions (e.g., snprintf, sleep), or the main function, and must be followed by an opening parenthesis indicating a function call. For variable names, candidates are identified by excluding function parameters, function names that are not keywords, preprocessor directives, library functions, or specific terms like 'argc' (representing the number of parameters passed to the main function) and 'argv' (representing the sequence or pointer of parameters passed to the main function). Additionally, the variable name is only standardized (e.g., VAR1) if it is not immediately followed by an opening parenthesis. This normalization process is illustrated in Figure1.

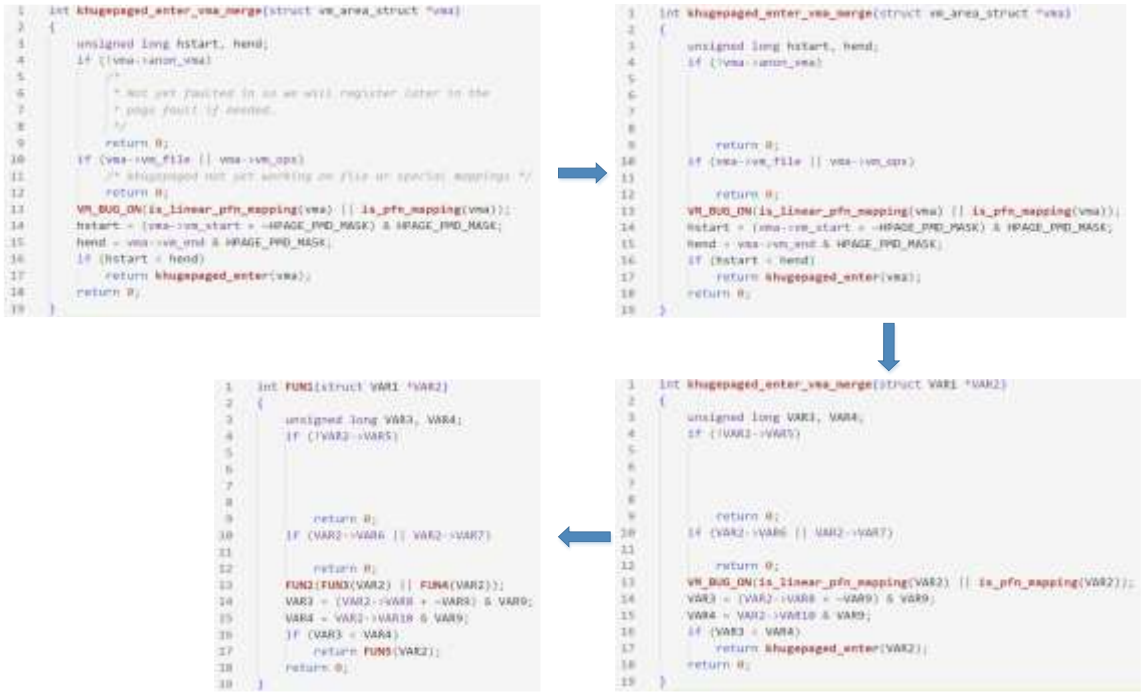


Figure 1. The Process of Code Standardization

B. Construction of Code Representation Graph

The code representation diagram is an effective method for visualizing code by clearly conveying its semantic, syntactic, and structural information. Among these, the PDG is a particularly expressive data structure that connects AST nodes through DDG and CDG. Using the Joern tool [27], binary (bin) files are generated, followed by the extraction of the CPG through the graph-for-funcs.sc script. This process generates both the AST and CFG, storing the resulting data in a JSON file.

The JSON format is chosen because the graph nodes are based on fine-grained AST elements rather than entire code statements. However, when analyzing source code, it is necessary to map these nodes to specific lines of code (for instance, diff files are line-based). By executing the command 'cpg.runScript("graph-for-funcs.sc").to String() |>.json', the graph is output as a JSON file, facilitating further analysis. The PDG, also generated by Joern, integrates AST and CFG structures, providing a more comprehensive representation of code features for vulnerability detection. The process of generating the PDG is illustrated in Figure2.

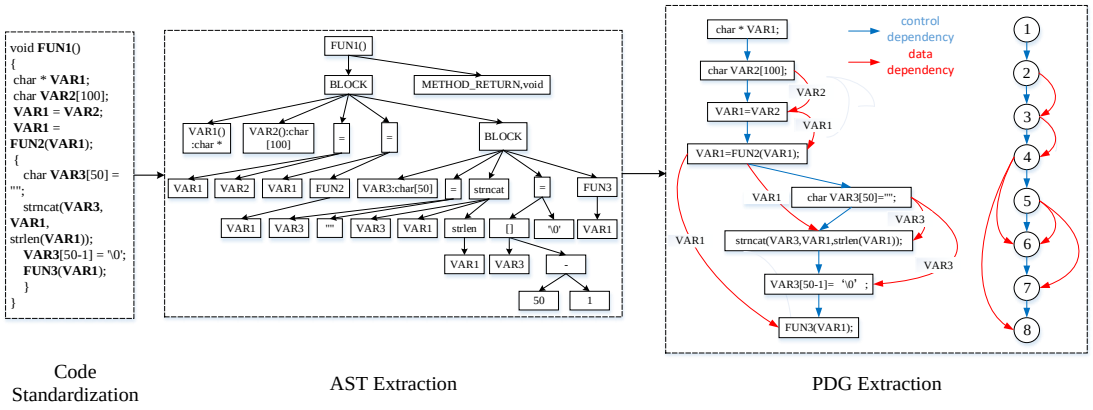


Figure 2. The Process of Generating the PDG

C. Slice of Code Representation Graph

A significant challenge in function-level vulnerability detection is the presence of a large number of vulnerability-independent noise statements, which can hinder the model's ability to effectively learn vulnerability features and ultimately degrade detection performance. Furthermore, the complexity of real-world software often results in program dependency graphs with numerous nodes and edges, leading to increased time and memory consumption during the training process. To address these challenges, this paper adopts a slice-level vulnerability detection approach, aiming to eliminate irrelevant information and enhance both detection accuracy and interpretability while minimizing resource overhead. By comparing vulnerable code with corresponding patched code, the study identifies four potential vulnerability-prone areas: pointers, arrays, expression operations, and sensitive APIs. These areas, referred to as "vulnerability focus points," represent key locations where vulnerabilities are most likely to occur.

1) Pointer Operations

Pointer operations can lead to dynamic memory errors, such as memory leaks, double-free errors, and null pointer dereferencing. To mitigate confusion associated with the use of the asterisk (*) in pointer declarations and avoid errors, this paper identifies specific node types in the code representation.

The node type 'Identifier' represents program entities such as variable names, functions, or classes. For instance, in the expression 'int* ptr', the term 'ptr' is classified as an identifier, and its data type is determined to be a pointer. Similarly, 'MethodParameterIn' nodes represent input parameters passed to methods or functions and are also identified as pointers when applicable. For example, in the function signature 'void foo(int* ptr)', 'ptr' is a method input parameter of pointer type. Additionally, 'FieldIdentifier' nodes are typically used to denote fields or member variables in a class or structure. Consider the following example:

```
struct MyStruct {
    int* ptr; // Pointer to an integer
};
```

In this case, the node representing 'ptr' is categorized as a pointer field. After identifying such node types, the system inspects the node and its type to check for the presence of the asterisk (*).

2) Array operation

Array operations often involve out-of-bounds reading or writing, with out-of-bounds writes potentially leading to memory overflow vulnerabilities. Specifically, selecting nodes classified as 'indirectIndexAccess' represents indirect index access, typically used to describe scenarios where arrays or collection elements are accessed indirectly through variables or expressions. For example, the following code illustrates a possible case of out-of-bounds access:

```
int arr [5] = {1, 2, 3, 4, 5};
int index = 2; int value = arr[index];
```

In this example, 'index' denotes an indirect access to the array. By analyzing such nodes in the code, the presence of array indexing symbols '[]' can be detected, allowing further checks for potential out-of-bounds access.

3) Expression Operations

Expression operations such as addition, subtraction, and multiplication can result in integer overflow. For instance, when performing arithmetic on 'int' types, if the result exceeds the representable range of the data type, overflow occurs. Division operations, on the other hand, may lead to division by zero errors.

The specific approach involves selecting nodes of type 'assignment', which represent assignment operations. If the node contains an equal sign ('='), the expression on the right side of the equal sign is extracted. Regular expressions are then used to match expressions that include arithmetic operations such as addition, subtraction, multiplication, and division (e.g., '((?:_|[A-Za-z])\w*(?:\s(?:\+|\-|*|\/)\s(?:_|[A-Za-z])\w*)+)' for strings like "a + b" or "x - y * z", where both operands start with a letter or underscore, followed by any number of letters, digits, or underscores). In cases where no equal sign is present, regular expressions are used to match division operations (e.g., '(?:\s\/\s(?:_|[A-Za-z])\w*\s)'), as division in

integer operations may trigger overflow or division-by-zero errors.

4) Sensitive API Function Operations

Improper use of functions that handle sensitive data can lead to various security vulnerabilities such as memory leaks, pointer errors, integer overflows, and buffer overflows. Examples include file handling functions (e.g., 'ifstream. open', 'ifstream.read*'), memory and pointer operations (e.g., 'xcalloc', 'IsBadReadPtr'), date and time functions (e.g., '_wctime_s', '_ctime64_s'), cryptographic functions (e.g., 'CC_SHA224_Update'), system calls and OS functions (e.g., 'chown', 'RegGetValue'), network communication functions (e.g., 'recvfrom', 'recv') and user input/output functions (e.g., 'getc', 'cin').

Starting from the four identified vulnerability focus points, the process involves traversing forward and backward along data dependency and control dependency edges, while preserving the original line numbers from the source code. These line numbers are then compared with those in the 'func_label.pkl' file. If the line numbers match, the slice is identified as containing a vulnerability and labeled as '1_'; otherwise, the slice is labeled as '0_', indicating no vulnerability in the slice.

As illustrated in Figure 3, slices 1 through 4 are derived using VAR1, VAR2, VAR3, and strncat as the slicing base points. Starting from these key points, forward and backward traversals along control and data dependency edges are conducted, recording the involved nodes and edges until no new nodes or edges emerge. The resulting subgraph of the program dependency graph, obtained through these steps, constitutes a program slice. Since the slice retains only nodes and edges that are dependent on the vulnerability focus points, it preserves the structural information of the original source code while eliminating irrelevant details.

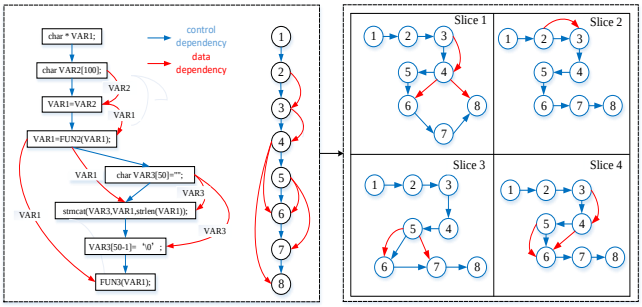


Figure 3. The Process of Slicing PDG

D. Extract Slice Features

Since the extracted code slices in this study are in an abstract graph format, they cannot be directly input into graph neural network-based vulnerability detection models. Therefore, key features from the graphs must be extracted to generate the corresponding feature vectors. The graph slices contain two types of features: code features within the nodes (referred to as node features) and the structural features of the graph.

Traditional Word2Vec models, which usually rely on local contextual information, are unable to capture long-range semantic relationships and global context. In contrast, BERT models excel in this area. BERT, through its bidirectional encoder pretraining, can deeply understand long-range dependencies within the context.

For node features, this study adopts an embedding representation approach, mapping tokens to integers and converting them into fixed-length vectors using a distributed representation technique. Specifically, the code within each node is treated as a sentence, tokenized into tokens, and embedded into a fixed-length vector. Node embeddings are achieved by combining the Word2Vec and BERT models.

Specifically, this study trains a pre-trained Word2Vec model using the token lists from all code slices. The pre-trained model is then applied to embed all nodes into vectors. The preprocessed slices are input into the Word2Vec model, which generates an $m \times n$ feature matrix Mf, where m represents the number of nodes in the slice, and n represents the dimension of the embedding vectors, which is set to 100 in this study. As shown in Figure 3, there are eight nodes in the graph, so the node feature matrix Mf has dimensions of 8×100.

For the BERT model, the pre-trained BERT model is utilized to embed the code within each node. The process involves using the 'BertTokenizer' to encode the tokens, followed by the 'BertModel' to generate context-sensitive dynamic word embeddings. Each node's code is first transformed into the input format accepted by BERT, which then captures the long-range dependencies and contextual information within the code snippets. The output of the BERT model for each node is a feature matrix with a shape of m×n, where n is 768 dimensions. As a result, the node feature matrix Mh in Figure 3 has a dimension of 8×768.

Finally, the word embeddings generated by Word2Vec and BERT are concatenated to form a combined vector of 8×868 dimensions, denoted as Mi.

For the graph structural features, this paper performs embedding representations of the edge relationships within the graph. Each edge can be represented as a triplet (source node, target node, edge type). Both the source and target nodes can be directly extracted from the program dependency graph, while the edge types are categorized into data dependency edges and control dependency edges. Taking Slice 1 as an example, it contains 9 edges, including 2 data dependency edges and 7 control dependency edges. Red edges represent data dependency, blue edges represent control dependency, and purple edges indicate both data and control dependencies. The output matrix AS represents the graph structure feature matrix, and its generation process is illustrated in Figure 4.

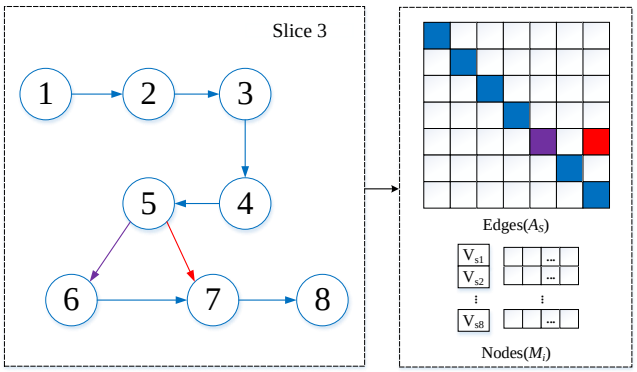


Figure 4. The Process of Extracting Features from the Slice Graph

E. Vulnerability Detection Model Based on GGNN

1) Figure Neural Network Module

This study transforms the source code into a graph structure that incorporates data dependencies and control dependencies. GNN help to further aggregate and propagate information updates, capturing both the structural and semantic information of the graph more effectively. Among GNN models, the GGNN is chosen for this work due to its enhanced ability to handle complex semantic and graph structure data by improving the network's long-term memory capacity. The principle behind GGNN involves aggregating information from a node and its neighboring nodes, then feeding the aggregated information and the current node into a GRU unit to obtain the updated state of the node. This process is repeated over several time steps, resulting in the final node representation for all nodes in the graph. As shown in the graph neural network module in Figure 5, after inputting the graph features gi(Mi,AS), GGNN embeds each node and its neighborhood into a new representation, transforming it into a slice feature matrix Mi' with dimensions m×n', where n' represents the final size of the slice feature. In this study, n' is set to 200, making the feature matrix Mi' of size 8×200.

For each node v_u in the graph, its initial feature vector $h_u^{(1)} = [m_u^T, 0]^T$ is constructed by concatenating the feature vector v_u with a zero padding. Setting T as the total number of time steps for neighborhood aggregation, each node communicates with its neighbors along the edges it depends on at each time step $t \le T$. The update formula is given by:

$$a_u^{(t)} = A_u^T (W_u [h_1^{(t)T}, ..., h_m^{(t)T}] + b) \quad (1)$$

where W_u represents trainable parameters, b is a bias term, and A_u^T denotes the adjacency matrix for the neighborhood of node v_u corresponding to edge type A_s. $a_u^{(t)}$ encapsulates the aggregated

information from node v_u 's neighbors through their interactions along the edges. This information is then combined with the node's current state through the aggregation function AGG, leading to the updated node state:

$$h_u^{(t+1)} = GRU(h_u^{(t)}, AGG(\{a_u^{(t)}\})) \quad (2)$$

This process continues iteratively, allowing the node's feature vector to evolve over time by incorporating information from neighboring nodes, until the final representation is obtained.

Vulnerability classification module

To perform graph-level vulnerability classification tasks, a feature set relevant to vulnerability characteristics is selected. Previous work [28] proposed using a classification pooling layer (SortPooling) after the graph convolutional layer to sort the output features, enabling the use of traditional neural networks for training and extracting useful features from the embedded slice vectors. In this paper, node features are first learned through GGNN layers, followed by one-dimensional convolutional and fully connected layers to capture features relevant to the graph classification task, enabling more effective classification. Specifically, skip connections are employed in the graph convolution and feature extraction phases, which help retain the details and semantic information from the original input data. This approach facilitates easier information propagation to deeper layers and prevents information loss. The process is expressed as follows.

$$\begin{aligned} H_3 = \mathrm{Relu}(&Conv3(\mathrm{Re\,lu}(Conv2(\mathrm{Re\,lu}(Conv1(X))) \\ &+ \mathrm{Re\,lu}(Conv1(X))))) \end{aligned} \quad (3)$$

The input graph structure data X is processed through the GNN layer to obtain node features Mi'. The first convolutional layer, Conv1, is used to extract initial node features, followed by Conv2, the second convolutional layer, which further captures higher-level node features. Conv3 represents the third convolutional layer and is responsible for extracting the final node features.

H3 denotes the node features after passing through all three convolutional layers. Finally, the obtained node features H3 are concatenated with the original input node features Mi, resulting in the feature matrix Mi'. This process is described as follows:

$$C_i = Contat(DeBatchify(H_3), DeBatchify(M_i)) \quad (4)$$

The function of DeBatchify is to restore node feature vectors into independent graph feature vectors when processing batch data, ensuring that each graph's data can be individually handled and analyzed.

In this paper, the classification pooling layer τ(M) is defined as follows:

$$\tau(M) = MaxPool(Relu(BN(Conv(M)))) \quad (5)$$

Here, Conv denotes the convolutional layer, BN represents the BatchNorm layer, ReLU indicates the activation function, MaxPool refers to the max-pooling layer, and M denotes a feature matrix. In this work, the node feature matrix Mi is concatenated with the corresponding slice feature matrix Mi1' to form a new matrix Mi1". τ-classification pooling operations are then applied separately to Mi1' and Mi1", resulting in outputs Y1 and Y2. These outputs are subsequently passed through fully connected layers with an output dimension of 2. The formulation for the weighted average and final output is presented as follows:

$$P = Sigmoid(Avg(MLPY(Y1) \cdot (MLPY(Y2))) \quad (6)$$

The fully connected layer performs a linear transformation on the feature matrix, followed by element-wise multiplication and averaging. The Sigmoid function is then applied to produce the binary classification probability output. P represents the binary classification result, consisting of two dimensions: the first dimension indicates the probability of no vulnerability, while the second dimension represents the probability of a vulnerability. The model outputs the final classification result by selecting the higher probability between the two. The model is trained using the CrossEntropyLoss function to correct misclassifications, along with the Adam optimization algorithm [29] with a learning rate of

0.0001 and a weight decay of 0.001 to update the parameters and b in the graph neural network module, as specified in Equation (1). After training, the model is used to determine whether new code slices contain vulnerabilities. The architecture of the vulnerability detection model is illustrated in Figure 5.

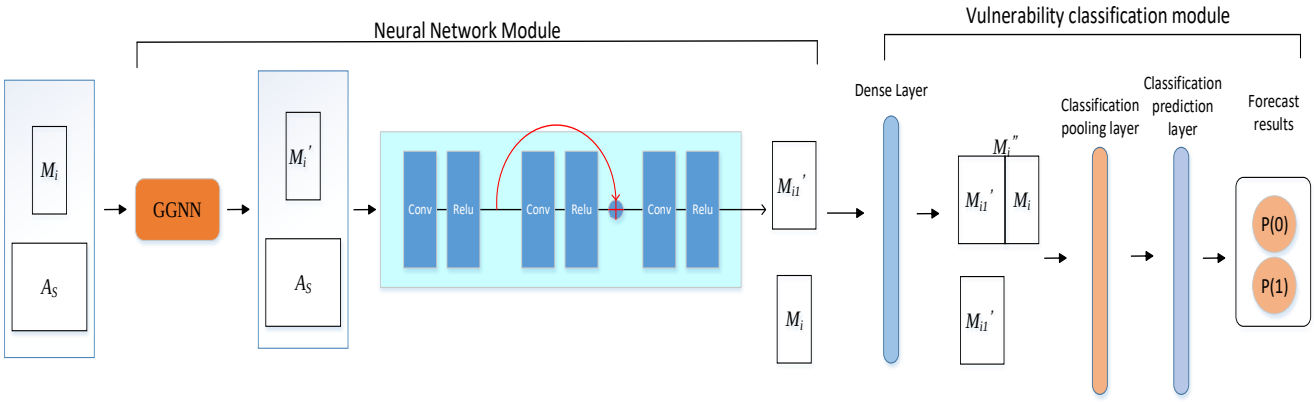


Figure 5. Vulnerability detection model architecture

IV. Experiment

A. Simulation parameter settings

This experiment is based on Python 3.8 to simulate and analyze the proposed algorithm, using the Pytorch 1.13 deep neural network framework and CUDA 11.6. The extraction of graphs in Joern is done using JDK17.0.11. Table I details the specific parameters used in the model during training.

TABLE I. TABLE TYPE STYLES

Parameter	Value
Loss Function	CrossEntropyLoss
Optimization Algorithm	Adam
Learning Rate	0.0001
Weight Decay	0.001
Batch Size	16
Training Epochs	500
Max Steps	10000

B. Training Results of the Proposed Network

This study utilizes the publicly available BigVul dataset, which includes 348 CVE (Common Vulnerabilities and Exposures) entries, consisting of 11,834 vulnerable functions and 253,096 non-vulnerable functions. From this dataset, 9,653 vulnerable functions and their corresponding 9,653 patched functions were selected for analysis. A total of 19,621 vulnerable code slices and 324,690 non-vulnerable slices were extracted. The difflib library was employed to generate the differential content between each vulnerable file and its respective patch file. Additionally, the specific lines of code containing vulnerabilities in each vulnerable file were recorded in the test_label.pkl file.

The Proposed Network achieved the highest test accuracy of 93.06%, precision of 92.22%, recall of 94.3%, and F1 score of 93.25%, all while maintaining stable training and test losses of 20% and 12.5%, respectively. These results indicate that the model is highly effective at accurately identifying vulnerabilities while maintaining a good balance between precision and recall. This suggests robust generalization and reliability in detecting both vulnerable and non-vulnerable code slices. Figure 6 shows the results of the model training in this article.

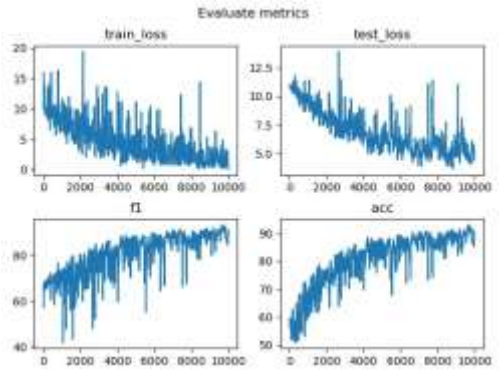


Figure 6. Results of the Model Training in the Proposed Network

C. Performance Comparison with Various Networks

Figure 7 presents a comparison of detection results using the proposed network under two approaches: one utilizing only Word2Vec and the other combining Word2Vec and BERT. The results for the model using only Word2Vec are as follows: Test Accuracy: 88.26, Precision: 87.28, Recall: 88.95, and F1 Score: 88.11. In contrast, the model combining Word2Vec and BERT achieved the following results: Test Accuracy: 93.06, Precision: 92.22, Recall: 94.30, and F1 Score: 93.25. These results demonstrate that integrating BERT significantly improves all evaluation metrics, highlighting its superior ability to capture contextual information and effectively extract deep semantic features.

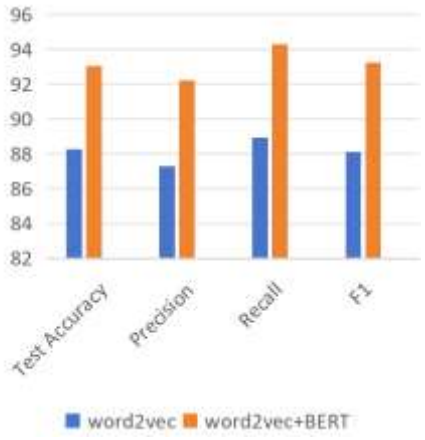


Figure 7. Ablation experiment

Through ablation experiments, we found that V1 (removing residual connections) achieved 91.13% accuracy, 88.65% precision, 94.65% recall, and 91.55% F1 score, but showed high training loss (100%). V2 (removing batch normalization) achieved 90.95% accuracy, 88.06% precision, 95.07% recall, and 91.43% F1 score, with training and test losses of 100% and 40%. V3 (replacing GGNN) performed the worst, with 71.99% accuracy, 72.7% precision, 71.83% recall, and 72.26% F1 score, alongside training and test losses of 20% and 50%. V_GIN, based on GIN layers, performed better than V1, V2, and V3, achieving 92.09% accuracy, 89.41% precision, 95.77% recall, and 92.49% F1 score, and had training and test losses of 15% and 40%. The performance comparison is illustrated in Figure 8, while the loss comparison is presented in Figure 9.

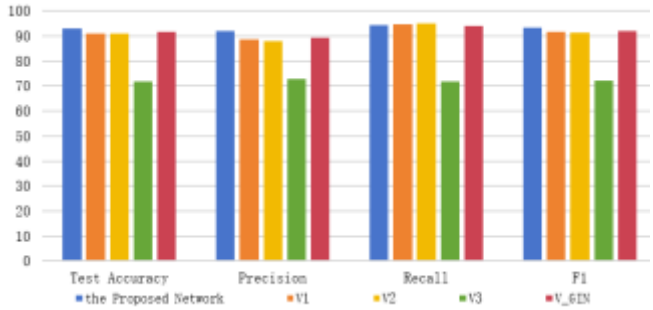


Figure 8. Ablation experiment

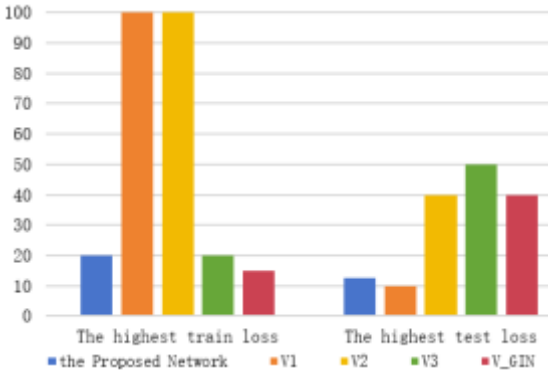


Figure 9. loss comparison

This study compares the proposed model with four deep learning-based vulnerability detection methods, as shown in Figure 10. TokenCNN [16], a token-based approach, treats source code as plain text and ignores semantic and structural information, leading to significant information loss and poor detection performance. StatementLSTM [31] improves on this by treating each line of code as a natural language sentence and embedding it into fixed-length vectors, reducing semantic loss. However, it also processes code as plain text, failing to preserve crucial syntactic and semantic details. Devign [22], a function-level method, uses code property graphs (CPGs) to capture comprehensive semantic and syntactic information. However, its inclusion of irrelevant nodes and edges, along with the absence of slicing techniques, hampers its ability to detect vulnerabilities effectively. Vuldetexp [28] simplifies code representation using slicing but relies solely on Word2Vec for embeddings, which limits its ability to extract rich code information. In contrast, the proposed model fully leverages

code semantics and structure while incorporating slicing techniques, achieving superior detection performance, robustness, and generalization compared to these methods.

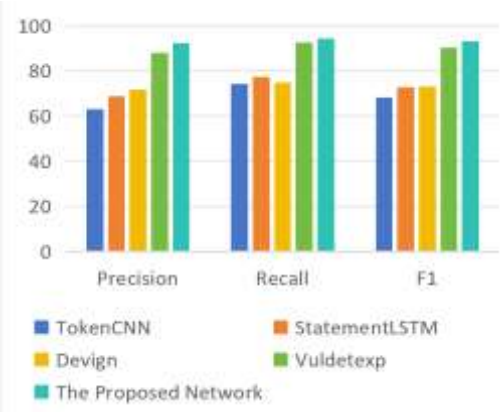


Figure 10. Performance comparison of different models under evaluation

V. Conclusions

This article provides a comprehensive overview of recent advancements in deep learning-based code vulnerability detection, categorizing the methods into sequence-based and graph-based approaches. It details the preprocessing steps involved, including code standardization, PDG (Program Dependency Graph) generation, PDG slicing, and the use of Word2Vec and BERT to extract comprehensive information from sliced graphs. Additionally, the study introduces a novel vulnerability detection method based on Graph Neural Networks (GNN), which extends traditional GGNNs by integrating skip connections, batch normalization, and advanced feature fusion mechanisms. Through ablation studies and comparisons with other deep learning-based methods, the proposed model demonstrates better performance in terms of accuracy, precision, recall, F1 score, and loss minimization. These findings highlight the effectiveness of skip connections in preserving features, batch normalization in enhancing training stability, self-attention mechanisms in capturing global dependencies, and BERT's ability to better extract features by leveraging contextual relationships in graph data, collectively enabling superior performance in vulnerability detection tasks.

Although the proposed model demonstrates significant improvements across several metrics, there are still areas that require further refinement. First, the model primarily analyzes code slices within single functions, making it challenging to handle the complex dependencies present in real-world vulnerabilities that span multiple functions. Future work should incorporate interprocedural analysis to enhance the detection of vulnerabilities involving multiple functions. Second, there is a severe imbalance between the number of vulnerable and non-vulnerable slices in the dataset, which can affect the model's generalization capability. Addressing this imbalance through techniques such as oversampling, undersampling, or the use of Generative Adversarial Networks (GANs) could help mitigate this issue. Additionally, the current model lacks interpretability, as it does not provide a clear indication of the specific code lines where vulnerabilities are detected. Future efforts should focus on integrating and improving tools like GNNExplainer to offer fine-grained explanations of the detection results, thereby enhancing the model's interpretability and practical utility.

Acknowledgment

The authors would like to thank the editor and reviewers for their constructive comments. This paper is supported in part by the Science and Technology Plan Project of Xi'an Beilin District(GX2214) under Grant, and in part by the Plan Project of the Xi'an Science and Technology(22GXFW0047) under Grant.

References

[1] Cnnvd. [EB/OL]. https: //www.cnnvd.org.cn.2023-7-19

[2] Hinton G E, Osindero S, Teh Y W. A Fast Learning Algorithm for Deep Belief Nets[J]. Neural Computation, 2006, 18(7): 1527-1554.

[3] Jacob D, Ming-Wei C, Kenton L, Kristina T, et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. [C], North American Chapter of the Association for Computational Linguistics, 2019, abs/1810.04805()

[4] Hoa K D, Truyen T, Trang P, Shien W N, John G, Aditya G, et al. Automatic feature learning for vulnerability prediction. [J], arXiv: Software Engineering, 2017, abs/1708.02368()

[5] Hoa K D, Truyen T, Trang P, Shien W N, John G, Aditya G, et al. Automatic Feature Learning for Predicting Vulnerable Software Components[J], IEEE Transactions on Software Engineering, 2021, 47(1): 67-85.

[6] Sanghoon Jeon, Huy Kang Kim. Autovas: An Automated Vulnerability Analysis System with A Deep Learning Approach[J], Computers & security, 2021, 106: 102308.

[7] Shigang L, Guanjun L, Lizhen Q, Jun Z, Olivier Y D V, Paul M, Yang X, et al. CD-VulD: Cross-Domain Vulnerability Discovery Based on Deep Domain Adaptation[J], IEEE Transactions on Dependable and Secure Computing, 2022, 19(1): 438-451.

[8] Thomas Shippey, David Bowes, Tracy Hall. Automatically identifying code features for software defect prediction: Using AST N-grams. [J], Information & Software Technology, 2019, 106(): 142-160.

[9] Junfeng Tian, Wenjing Xing, Zhen Li. BVDetector: A Program Slice-based Binary Code Vulnerability Intelligent Detection System[J], Information & Software Technology, 2020, 123(): 106289.

[10] Song Wang, Taiyue Liu, Lin Tan. Automatically learning semantic features for defect prediction. [J], Proceedings - International Conference on Software Engineering. International Conference on Software Engineering, 2016: 297-308.

[11] Fabian Y, Christian W, Hugo G, Konrad R, et al. Chucky: exposing missing checks in source code for vulnerability discovery[J], Computer Science, 2013: 499-510.

[12] Thong H, Hoa K D, Yasutaka K, David L, Naoyasu U, et al. DeepJIT: an end-to-end deep learning framework for just-in-time defect prediction[C], IEEE Working Conference on Mining Software Repositories, 2019: 34-45.

[13] Luca Pascarella, Fabio Palomba, Alberto Bacchelli. Fine-grained just-in-time defect prediction. [J], Journal of Systems and Software, 2019, 150(): 22-36.

[14] Yan, X. D. Research on Software Vulnerability Detection Technology Based on Static Taint Analysis and Deep Learning [Master's thesis, Harbin Institute of Technology]. DOI: 10.27061/d.cnki.ghgdu.2021.003610.

[15] Wu F, Wang J, Liu J, Wang W. Vulnerability detection with deep learning//Proceedings of the International C onference on Computer and Communications. Chengd u, China, 2017: 1298-1302.

[16] Rebecca L R, Louis K, Lei H H, Tomo L, Jacob A H, Onur O, Paul M E, Marc W M, et al. Automated Vulne rability Detection in Source Code Using Deep Represe ntation Learning[J], 2018 17th IEEE International Conf erence on Machine Learning and Applications (ICML A), 2018, abs/1807.04320: 757-762.

[17] Han Y, Senlin L, Limin P, Yifei Z, et al. Han-Bsvd: A Hierarchical Attention Network for Binary Software V ulnerability Detection[J], Computers & security, 2021, 108: 102286.

[18] Li Z, Zou D, Xu S, et al. VulDeeLocator: A Deep Learning-based Fine-grained Vulnerability Detector[J]. IEEE Transactions on Dependable and Secure Computing, 2022, 19(4): 2821-2837.

[19] Li Z, Zou D, Xu S, et al. VulDeePecker: A Deep Learning-Based System for Vulnerability Detection[C]//Proceedings 2018 Network and Distributed System Security Symposium. 2018.

[20] Zou D, Wang S, Xu S, et al. μDeePecker: A Deep Learning-Based System for Multiclass Vulnerability Detection[J]. IEEE Transactions on Dependable and Secure Computing, 2021, 18(5): 2224-2236.

[21] Li Z, Zou D, Xu S, et al. SySeVR: A Framework for Using Deep Learning to Detect Software Vulnerabilities[J]. IEEE Transactions on Dependable and Secure Computing, 2022, 19(4): 2244-2258.

[22] Zhou, Y, Liu, S, Siow, J, Du, X, Liu, Y, et al. Devign: Effective Vulnerability Identification by Learning Comprehensive Program Semantics via Graph Neural Networks[J], Advances in neural information processing systems, 2019, 32(): 10197-10207.

[23] Wang H, Ye G, Tang Z, et al. Combining Graph-Based Learning with Automated Data Collection for Code Vulnerability Detection[J]. IEEE Transactions on Information Forensics and Security, 2021, 16: 1943-1958.

[24] Zheng W, Jiang Y, Su X. Vu1SPG: Vulnerability Detection Based on Slice Property Graph Representation Learning[J], IEEE International Symposium on Software Reliability Engineering, 2021.

[25] Cao S, Sun X, Bo L, et al. MVD: Memory-Related Vulnerability Detection Based on Flow-Sensitive Graph Neural Networks[C]//Proceedings of the 44th International Conference on Software Engineering. 2022: 1456-1468.

[26] Zou D, Hu Y, Li W, Wu Y, Zhao H, Jin H. mVulPreter: A Multi-Granularity Vulnerability Detection System with Interpretations[J], IEEE Transactions on Dependable and Secure Computing, 2022, PP (99): 1-12.

[27] Fabian Y, Nico G, Daniel A, Konrad R, et al. Modeling and Discovering Vulnerabilities with Code Property Graphs[C], IEEE Symposium on Security and Privacy, 2014: 590-604.

[28] Hu Yutao, Wang Suyuan, Wu Yueming, et al. A Graph Neural Network-Based Method for Slice-Level Vulner a-bility Detection and Explanation[J]. Journal of Softw are,2023,34(06): 2543-2561. DOI:10.13328/j.cnki.jos.0 06849.

[29] Kingma, Diederik P., and Jimmy Lei Ba. Adam: A Method for Stochastic Optimization[J]. International Conference on Learning Representations (ICLR), 2014, abs/1412.6980.

[30] Cheng X, Wang H, Hua J, et al. DeepWukong: Statically Detecting Software Vulnerabilities Using Deep Graph Neural Network[J]. ACM Transactions on Software Engineering and Methodology, 2021, 30(3): 1-33.

[31] Lin G, Xiao W, Zhang J, et al. Deep learning-based vulnerable function detection: A benchmark. In: Proc. of the 21st Int ′ l Conf. on Information and Communications Security (ICICS 2019). 2019. 219-232.

[32] Yang Y, Li G. On the Code Vulnerability Detection Based on Deep Learning: A Comparative Study[J]. IEEE Access, 2024.

Pavement Damage Recognition Based on Deep Learning

Mingbo Ning
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: 450156598@qq.com

Shengquan Yang
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, China
E-mail: xaitysq@126.com

***Abstract*—Road surface disease detection is a vital component of road maintenance. Traditional deep learning-based detection methods face challenges such as low detection accuracy, high false alarm rates in complex scenarios, and significant missed detection rates for small targets like potholes. To address these limitations, this paper proposes an improved pavement disease detection algorithm based on RT-DETR. First, a lightweight backbone network named LMBANet is constructed by integrating DRB and ADown modules. This network enhances feature extraction capabilities without increasing computational overhead during inference, preserving local details of low-level features while expanding the receptive field to capture long-range semantic information and reduce false detection of diverse defects in complex scenes. Second, an small-target enhanced feature pyramid network is designed using SPDConv and OmniKernel. By feeding large-scale feature maps extracted by the backbone into a feature fusion layer and enhancing multi-scale feature representation through EFKM, this network resolves the high missed detection rate of small targets in the original model. Experimental results demonstrate that on the RDD2020 dataset, the improved network achieves an mAP of 69.2%, representing a 2.1 percentage point improvement over the original network, while simultaneously reducing parameters and computational costs.**

Keywords-Deep Learning; Road Surface Disease Detection; RT-DETR; Lmbablock; STEP

I. INTRODUCTION

Roads are critical components of the transportation system, with highway construction playing a particularly vital role in infrastructure development. Highway transportation significantly facilitates public travel and accelerates socioeconomic progress. However, pavement health issues can severely impact traffic safety. If maintenance is delayed until obvious pavement damage occurs, repair costs will escalate dramatically. Therefore, early detection and repair of potholes and cracks using intelligent inspection technologies are essential for ensuring transportation safety and reducing long-term maintenance expenses.

Early pavement damage identification and assessment methods primarily relied on manual inspections conducted by road maintenance workers. These workers would patrol the road network, visually inspecting and manually measuring various damage parameters to evaluate the overall pavement deterioration. Although this human-based approach offers simplicity and relatively high accuracy, it suffers from several significant drawbacks: the labor-intensive process is time-consuming and inefficient, often causing urban traffic congestion during inspections, which adversely impacts transportation efficiency and poses potential safety hazards. Consequently, manual inspections have gradually been replaced by specialized pavement inspection vehicles equipped with professional Charge-Coupled Device (CCD) cameras. These vehicles enable quantitative assessment of road defects through continuous video recording without disrupting normal traffic flow. However, they still require manual image processing for damage analysis, and their high operational costs fail to resolve the substantial consumption of human and financial resources.

With the remarkable success of deep learning technology, computer vision approaches have been widely adopted for pavement damage detection tasks. Current mainstream object detection models, however, struggle to balance computational complexity with detection performance. Models with high computational complexity face

deployment challenges in real-world scenarios, while lightweight models with reduced computations often exhibit insufficient detection accuracy, particularly showing susceptibility to false positives and missed detections under complex environmental conditions. These limitations hinder their ability to meet practical engineering requirements. To address these challenges, this paper proposes an enhanced model based on RT-DETR (Real-Time Detection Transformer), aiming to optimize both computational efficiency and detection reliability in pavement damage identification.

II. RELATED WORK RESEARCH

In recent years, with advancements in artificial intelligence and computer hardware technologies, scholars have progressively applied object detection models such as Faster R-CNN, YOLO, and DETR to pavement damage detection. These algorithms enable automatic identification of damaged road areas through single-image input while achieving satisfactory detection performance. Li et al. [1] employed Faster R-CNN to analyze 5,966 road defect images captured from diverse angles and distances. Experimental results demonstrated the model's robust detection capability under varying illumination conditions, effectively recognizing five categories of road defects: transverse cracks, longitudinal cracks, potholes, alligator cracks, and manhole-related defects.

The YOLO series of algorithms achieve extremely fast inference speeds while maintaining high detection accuracy, and their robust real-time detection capabilities have made them widely adopted in pavement damage recognition. Joseph Redmon et al. [2] introduced a feature pyramid network in YOLOv3 to leverage multi-scale feature maps for improving recognition accuracy of targets of varying sizes. Duan et al. [3] further enhanced cross-scale feature extraction by integrating a Bi-directional Feature Pyramid Network (BiFPN).

The success of Transformer models in natural language processing has demonstrated the exceptional capability of attention mechanisms in integrating global contextual semantic information. Researchers began exploring their applications in computer vision. Dosovitskiy et al. [4] proposed the Vision Transformer (ViT), a deep learning model specifically designed for computer vision tasks using self-attention mechanisms. ViT processes input images by dividing them into patchembeding, learning global contextual information through self-attention, and subsequently passing these features to fully connected layers for classification or regression tasks. However, ViT's global attention mechanism requires computing pairwise relationships between all image patches, resulting in quadratic computational complexity ($O(N^2)$) that poses challenges for high-resolution images and large-scale datasets.

Facebook AI [5] introduced DETR (Detection Transformer) in 2020 as an end-to-end global detection framework. DETR employs a CNN backbone for feature extraction followed by Transformer encoder-decoder layers for prediction. It replaces anchor generation with learnable object queries and utilizes a bipartite matching-based loss function to enforce one-to-one prediction matching, eliminating non-maximum suppression (NMS). Building upon DETR, Zhu et al. [6] proposed Deformable Attention to address the $O(N^2)$ complexity of standard attention, resolving slow convergence and high feature map dependency. Chen et al. [7] developed Group DETR, which employs multiple object queries to retain end-to-end inference advantages while accelerating convergence through one-to-many supervision. DINO [8] enhances detection robustness via contrastive denoising to reduce anchor dependency and improve occluded object recognition. Co-DETR [9] implements a collaborative hybrid training scheme with auxiliary detectors like ATSS and Faster R-CNN, enriching supervision signals for small object detection. MFDS-DETR [10] introduces a hierarchical semantic FPN (HS-FPN) to optimize multi-scale feature fusion, significantly boosting small target detection accuracy.

In 2023, Baidu's PaddlePaddle team [11] introduced RT-DETR (Real-Time Detection Transformer), a highly practical industrial-grade detector featuring an efficient hybrid encoder. This architecture combines an Attention-based Intra-scale Feature Interaction Module for contextual refinement and a CNN-based Cross-scale Feature-

fusion Module for multi-level integration, achieving real-time performance through computational redundancy reduction while maintaining detection precision.

Figure 1. Road surface damage dataset under different conditions

However, most existing approaches primarily predict pavement crack defects under conventional conditions, demonstrating limited robustness in complex environmental scenarios. As illustrated in Figure 1, these challenging scenarios include shadow interference, rainy conditions, color segmentation ambiguities, dense defect distributions, and pothole clusters. Current object detection algorithms generally suffer from three critical limitations: Similarity between defect features and background textures frequently causes false positives; The spatial continuity and linear characteristics of cracks often lead to misclassification alligator cracks as other defect types; Significant scale variations between defects result in frequent missed detections of small targets like potholes. To address these challenges while maintaining real-time detection capabilities, this paper proposes an enhanced RT-DETR-based model.

III. METHODS

A. RT-DETR Network

RT-DETR is a Transformer-based real-time object detection model that employs an HybriDencoder to reduce computational redundancy through decoupled intra-scale interactions and cross-scale fusion, while maintaining detection accuracy. By eliminating post-processing operations like non-maximum suppression (NMS), the algorithm achieves enhanced inference efficiency and fully leverages end-to-end advantages. Given the requirements for low computational overhead and high real-time performance in pavement defect detection tasks, this paper selects the relatively lightweight RT-DETR-r18 as the baseline model. The overall network architecture is illustrated in Figure 2.

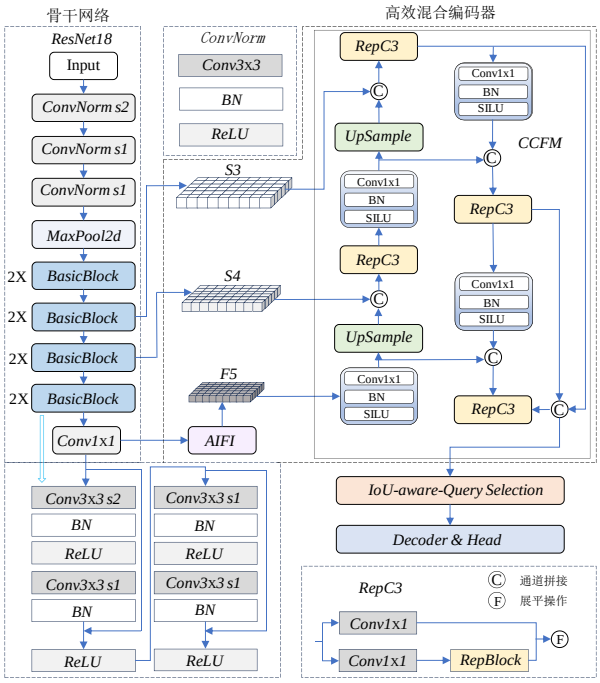


Figure 2. RT-DETR-r18 model structure

The model comprises three core components: Backbone, HybridEncoder, TransformerDecoder. RT-DETR adopts ResNet18 [12] as its backbone - a classical deep residual network characterized by shallow architecture and robust performance. Through residual blocks implementing cross-layer connections, ResNet18 effectively mitigates vanishing gradient issues. The hybrid encoder consists of two specialized modules: the Attention-based Intra-scale Feature Interaction (AIFI)

module and the CNN-based Cross-scale Feature Fusion (CCFM) module.

The input image first undergoes multi-scale feature extraction through the backbone network. High-level semantic features from the S5 layer are then flattened and processed by the AIFI module with positional encoding. Multi-head attention mechanisms execute intra-scale feature interactions within AIFI, with the output subsequently reshaped into 2D features (denoted as F5) for cross-scale fusion. The CCFM module inserts convolutional Fusion Blocks into the fusion path to integrate adjacent-scale features. Finally, IoU-aware queries select fixed-length features from the encoder's output sequence as initial object queries for the decoder. These queries are optimized through auxiliary pre-detection heads to generate final class predictions and bounding boxes. The representation process is:

$$Q = K = V = Flatten(S5) \tag{1}$$

$$F5 = reshape\left(Attn(Q,K,V)\right) \tag{2}$$

$$Output = CCFM\left(\{S3,S4,F5\}\right) \tag{3}$$

Among them, flatten denotes the flattening operation, Attn refers to multi-head self-attention, and reshape represents the process of restoring features to the same shape as S5.

B. Improving RT-DETR Network

The improved model utilizes a more lightweight network compared to ResNet18 for shallow feature extraction, achieving a larger effective receptive field to capture long-range semantic information. The input image generates four-scale feature maps S2, S3, S4, and S5 through the backbone network. Among them, the S5 feature is encoded into F5 within the original model's AIFI module. S2, S3, S4, and F5 are then fed into an enhanced small-object feature pyramid fusion network. The upsampled F5 feature map is concatenated with the S4 feature map along the channel dimension. The resulting output is upsampled again and concatenated with the S2 feature map processed by SPDConv along the channel dimension. The final output undergoes EFKM processing to generate a feature map containing small-scale information. Through a series of multi-scale feature fusions, the model ultimately produces a comprehensive feature map with effective information across all scales, which is then input into the decoder.

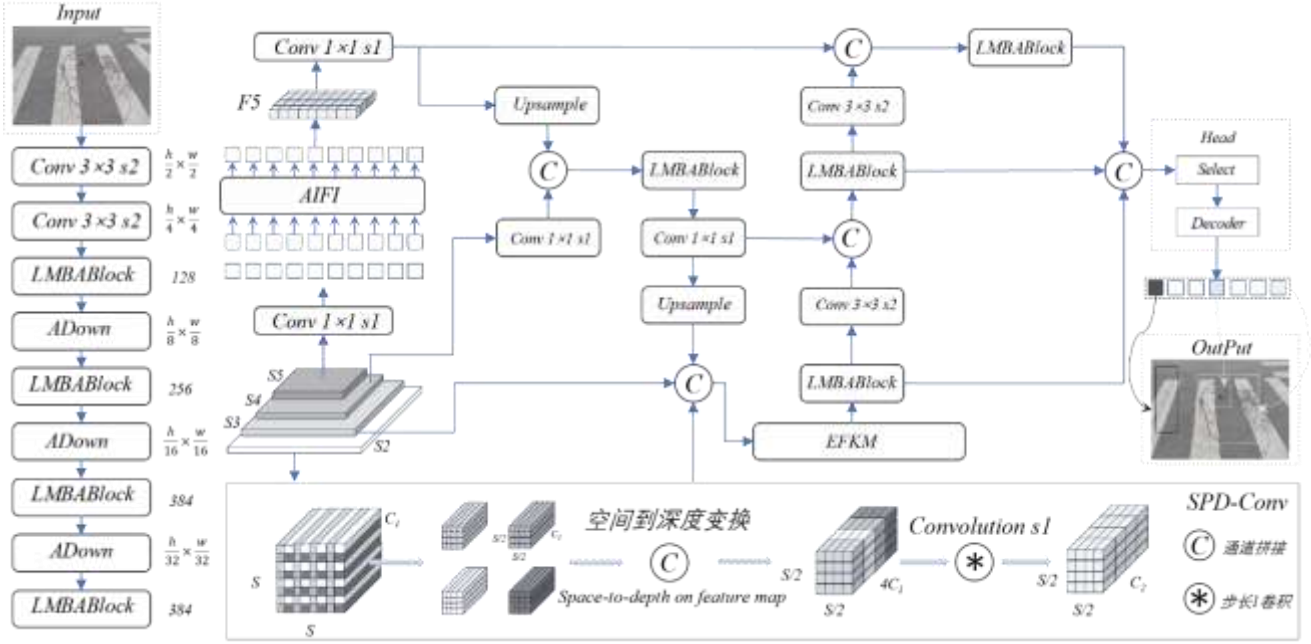


Figure 3. Improved RT-DETR model structure

C. LMBANet

The coexistence of multiple pavement defects often leads to model misdetections across various damage types. For instance, in complex scenarios, there exists significant similarity between alligator cracks and transverse cracks, as illustrated in Figure 4. Such cases may cause misclassification between crack types, subsequently affecting maintenance crews' root cause analysis and targeted repair strategies. To address this challenge, we integrate GELAN with Dilated convolution principles to design a Long-range feature extraction backbone network.

Figure 4. Diagrams of different types of cracks

GELAN [13] is an efficient aggregation network combining CSPNet architecture with gradient path optimization, enabling effective propagation and integration of multi-level feature information. The network partitions input feature tensors into two streams: one preserves original features through identity mapping, while the other undergoes multi-layer convolutional operations to extract higher-level abstractions. These streams are concatenated through multi-stage channel-wise fusion.

The Dilated Re-param Block (DRB) [14] enhances feature representation through a re-parameterization mechanism based on dilated convolutions. During training, the module employs a 7×7 non-dilated convolution layer parallel with three dilated convolutional branches {kernel sizes=5,3,3, dilation rates=1,2,3}. Outputs from these branches are batch-normalized and aggregated additively. During inference, re-parameterization converts the entire structure into an equivalent single non-dilated convolution layer, eliminating computational overhead from auxiliary branches.

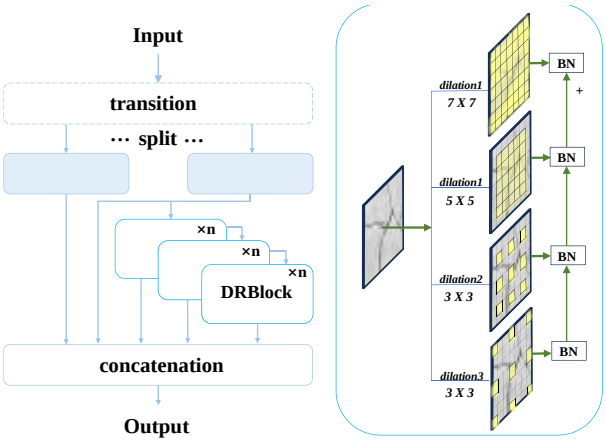


Figure 5. Structure of LMBA module

We integrate DRB into GELAN's branch pathways to create a Long-Road Multi-branch Aggregation Block (LMBABlock), as detailed in Figure 5. Replacing original feature extraction modules, DRB-enhanced branches capture multi-receptive-field features. The aggregated multi-scale features from parallel branches enable long-range semantic understanding. The input features first undergo channel and spatial dimension adjustment through a convolutional layer, before being processed by the LMBABlock to extract multi-scale features with large receptive fields. These features are subsequently downsampled through the Adown [14] module - an innovative downsampling component that splits the input features into two parallel paths: one path employs stride-3 convolution to preserve original structural information, while the other utilizes max pooling to extract salient features. Through the stacked configuration of LMBABlock and ADown modules, the complete backbone network architecture is constructed, as shown in the left portion of Figure 5.

D. STEP

Potholes, as typical small-scale targets in pavement damage detection, often suffer from information degradation during feature propagation from shallow to deep layers. Due to the inherent locality of feature mapping and varying receptive field scales across network depths, fine-grained details in abstract feature maps are progressively weakened, leading to frequent missed detections of small targets. Figure 6(a) illustrates the original cross-scale fusion network in RT-DETR, which

constructs top-down and bottom-up feature pyramid pathways for multi-scale interactions. However, this interaction initiates from the P3 detection layer, inherently limiting the model's capacity to preserve small-scale semantic information. Traditional improvement approaches, as shown in Figure 6(b), address this by adding a P2 small-target detection layer, but inevitably introduce excessive computational overhead. To resolve this dilemma, we propose an small-target enhanced feature pyramid architecture specifically optimized for small targets, depicted in Figure 6(c). The P2 feature map first undergoes SPDConv [15] to enrich small-target representations, then employs our improved EFKM (Efficient Full Kernel Module) derived from OmniKernel [16] Module for efficient feature consolidation while maintaining computational efficiency.

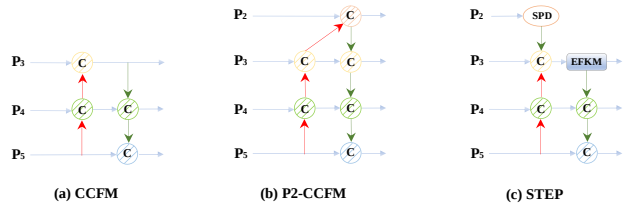


Figure 6. Comparison of feature Pyramid net

The SPDConv module comprises a Space-to-Depth (SPD) layer followed by a non-strided convolution layer, with its architectural details illustrated in the lower section of Figure 3. The SPD layer reduces the spatial dimensions while expanding the channel dimensions of the input feature map, effectively preserving spatial information without loss. After processing through SPDConv, the resulting P2-level feature maps undergo cross-scale fusion with P3 and P4 features within the EFKM to integrate multi-resolution representations.

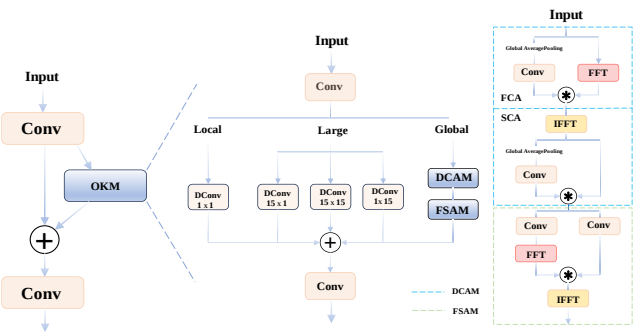


Figure 7. Structure of EFKM module

The EFKM (Efficient Full Kernel Module) architecture is illustrated in Figure 6. Given input features $X \in RC \times H \times W$ from the OKM (Omni-Kernel Module), the features undergo 1×1 convolutional processing before being distributed to three parallel branches: the local branch, large kernel branch, and global branch, which collectively enhance multi-scale representations. The outputs from these branches are aggregated through element-wise summation and subsequently modulated by another 1×1 convolution.

The large kernel branch employs a computationally efficient large-kernel depthwise convolution ($K \times K$) to capture extensive receptive fields. Complementing this, parallel $1 \times K$ and $K \times 1$ depthwise convolutions are utilized to extract strip-shaped contextual information. To address the limitation of large kernels in achieving global coverage, the global branch incorporates a Dual-domain Channel Attention Module (DCAM) and a Frequency-based Spatial Attention Module (FSAM). For input features $X_{Global} \in R^{C \times H \times W}$, the DCAM first applies Frequency Channel Attention (FCA), expressed as:

$$X_{FCA} = IF\left(F\left(X_{Global}\right)\right) Conv \{ \left\{GAP\left(X_{Global}\right)\right\} \quad (4)$$

Where F and IF denote Fast Fourier Transform (FFT) and its inverse, respectively. The operator

$\odot$ represents element-wise multiplication, while GAP and Conv indicate global average pooling and 1×1 convolution. Optimized features from FCA are then fed into the Spatial Channel Attention (SCA) module as described in equation:

$$X_{DCAM} = X_{FCA} \{ Conv\left\{GAP\left(X_{FCA}\right)\right\} \quad (5)$$

Here, XDCAM represents the output of DCAM. Following channel-wise enhancement, FSAM performs fine-grained spectral refinement in the spatial dimension through frequency-based attention mechanisms, formally defined as:

$$X_{FSAM} = IF(W1 \{ W2) \quad (6)$$

$$W1 = F\left(Conv\{ X_{DCAM} \}\right) \quad (7)$$

$$W2 = Conv\left\{ X_{DCAM} \right\} \quad (8)$$

Where W1 and W2 derive from frequency-domain and spatial-domain transformations of XDCAM, respectively. This enables the module to prioritize frequency components carrying critical semantic information. In addition to the large kernel branch for extended receptive fields and the global branch for full-scale coverage via dual-domain processing, a lightweight local branch supplements local detail preservation through a simple 1×1 depthwise convolution.

IV. EXPERIMENTS

A. Experimental Environment

Table I shows the experimental environment in this paper, which is based on the Ubuntu 18.04 operating system, the graphics card model is RTX4090D, and the memory is 24GB. The experiment basically uses the parameters recommended by RT-DETR, builds the model based on Python3.9 and Pytorch1.13.1 framework, and uses the standard SGD optimizer, with batch-size set to 8 and epochs set to 150.

TABLE I. EXPERIMENTAL ENVIRONMENT

Experimental environment	Version
CPU	Intel Xeon Platinum 8352V
GPU	NVIDIA GeForce RTX4090D
Language	Python3.9
Deep Learning Framework	Pytorch1.13.1
CUDA	11.6.0

B. Dataset

In this experiment, we utilized the publicly available RDDC2020 [17] dataset provided by the Global Road Damage Detection Challenge. The original RDD2020 dataset comprises 26,336 road images collected from India, Japan, and the Czech Republic. To better align with domestic road surface environments, a subset of 9,600 images demonstrating similar characteristics to Chinese pavement conditions was carefully selected for our study. Following standard experimental protocols, the dataset was partitioned into training and testing sets, with 80% allocated for training purposes and the remaining 20% reserved for testing. The quantitative distribution of different damage category labels is systematically presented in Table 2, illustrating the sample statistics across various defect types.

TABLE II. DISEASE CATEGORY

Category	Train Set	Test Set
D00(Longitudinal cracks)	7419	876
D10(Transverse cracks)	5702	636
D20(Alligator cracks)	6244	689
D40(Potholes)	2316	248

C. Evaluation Metrics

In this study, the following evaluation metrics were adopted: precision (P), recall (R), average precision (AP), mean average precision (mAP), model parameter count, and computational complexity measured in Giga Floating-point Operations Per Second (GFLOPs). The mAP metric, one of the most widely used benchmarks for object detection performance, is derived from the precision-recall relationship. Its calculation procedure follows the equations below [18]:

$$P = TP/(TP+FP) \tag{9}$$

$$R = TP/(TP+FN) \tag{10}$$

$$AP = \int_0^1 P(R)d(R) \tag{11}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{n} APi \tag{12}$$

Where TP denotes true positives (correctly detected positive samples), FP represents false positives (negative samples erroneously classified as positive), FN indicates false negatives (positive samples misclassified as negative), N is the total number of damage categories, and APi denotes the detection accuracy for the ii-th category, calculated through precision-recall integration.

Parameter count quantifies model size, while computational complexity (GFLOPs) evaluates the arithmetic operations required during inference.

Models with lower parameter counts and computational demands are prioritized for lightweight deployment scenarios, as they reduce hardware resource requirements while maintaining detection efficacy.

D. Algorithm verification results

The detection performance comparison between RT-DETR and its improved variant on the test set is systematically summarized in Table 3.

TABLE III. COMPARISON BEFORE AND AFTER IMPROVEMENT

Algorithm	Pars/M	FLOPS/G	FPS/f/s	mAP/%
RT-DETR	19.8	57.3	69	67.1
Improved RT-DETR	14.6	45.2	60	69.2

As evidenced by the quantitative results, the enhanced model demonstrates superior detection accuracy across all damage categories, achieving a 3.8 percentage point improvement for small-target D40 potholes, along with 3.2 and 2.2 percentage point gains for easily confounded D10 and D20 defects under complex scenarios. The overall mean average precision (mAP) shows a marked enhancement, while model parameter count and computational complexity are reduced by 29% and 10%, respectively, compared to the baseline. Although the frames per second (FPS) slightly decreases from 69 to 60, this operational speed remains well above the 30 FPS threshold required for practical road damage detection systems deployed on vehicular or drone platforms. Although the frames per second (FPS) slightly decreases from 69 to 60, this operational speed remains well above the 30 FPS threshold required for practical road damage detection systems deployed on vehicular or drone platforms.

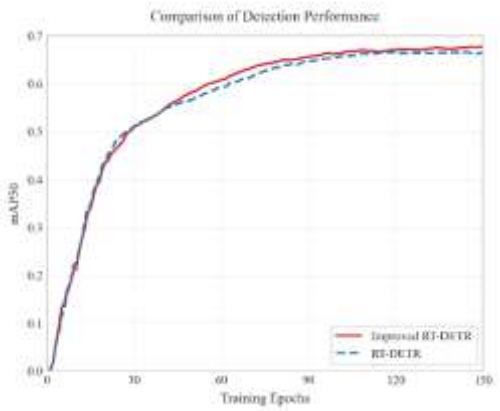


Figure 8. Comparison chart of mAP during training

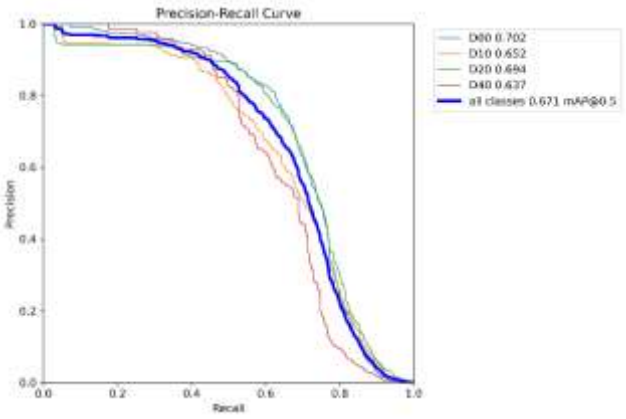


Figure 9. Average precision of each label in RT-DETR

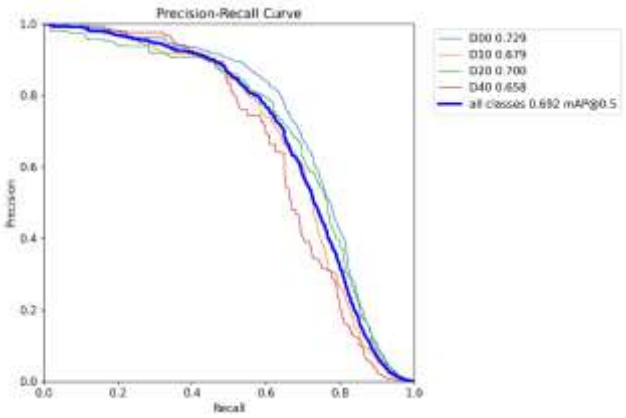


Figure 10. Average precision of each label in Improved RT-DETR

Figures 8-10 provide detailed performance analyses: Figure 8 contrasts the mAP evolution during training between the original and improved models, while Figures 9 and 10 visualize their precision-recall characteristics on the test set. The

baseline RT-DETR's suboptimal detection of transverse cracks and potholes stems from its limited receptive field, which frequently misclassifies transverse cracks as reticular counterparts. In contrast, the enhanced architecture strategically integrates local texture patterns with global semantic contexts through multi-scale feature fusion, thereby acquiring significant advantages in small-target recognition and spatial relationship modeling.

Figure 11 presents the detection outcomes of the algorithm before and after improvement in different scenarios of the selected dataset. From left to right, the scenarios are normal conditions, color interference, dense diseases, dense small targets, and low - light conditions. As can be seen from Figure 11, the algorithm improved by introducing the enhanced small-object feature pyramid network managed to identify the tiny potholes that RT - DETR failed to detect in the dense small - target scenario. Moreover, in the dense - disease and color - interference scenarios, the improved algorithm did not mix up transverse cracks with networked cracks.

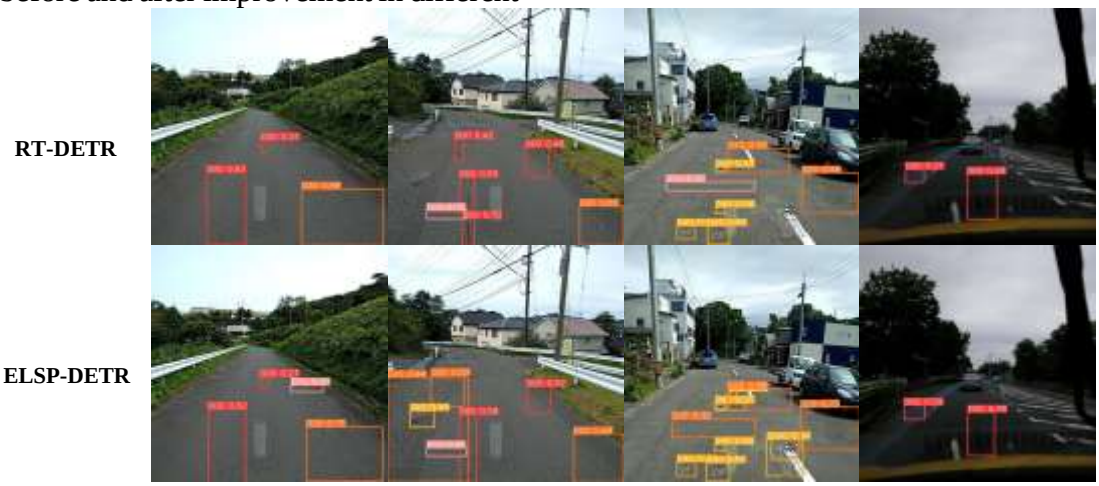


Figure 11. Visual comparison of test results

E. Ablation experiment

TABLE IV. COMPARISON BEFORE AND AFTER IMPROVEMENT

Experiments	LMBAN	STEP	Pram/M	FLOPs/G	mAP/%
I			19.8	57.3	67.1
II	✓		12.8	41.9	68.3
III		✓	20.5	59.5	68.9
IV	✓	✓	14.6	45.2	69.2

The model improvement is based on the RT-DETR architecture. To validate the effectiveness of each modification, ablation experiments evaluating detection accuracy and computational resource consumption were conducted using the dataset adopted in this study with results presented in Table 4.

The original RT-DETR model's performance metrics are shown in the first experimental configuration. Replacing its backbone network improved model accuracy by 1.2 percentage points while reducing parameters by 35% and computational cost by 26%, demonstrating efficiency gains without sacrificing detection capability. Substituting the original CCFM structure with STEP increased mAP by 1.8 percentage points compared to the baseline, indicating enhanced representation of small-scale features despite higher computational requirements. Combining both modifications achieved 2.1 percentage point mAP improvement over the original model while reducing parameters by 26% and computational cost by 21%.

F. Comparison experiment

To further validate the superiority of the improved algorithm for pavement disease detection, comparative experiments were conducted between

the proposed algorithm and conventional object detection algorithms. All experiments were performed under identical software and hardware environments using the same dataset, with results presented in Table 5.

Table 5 demonstrates that the improved algorithm achieves the highest accuracy among all compared methods. [19-20] Meanwhile, its parameter count and computational cost are significantly lower than those of other mainstream algorithms, enabling better adaptability of the model in edge device environments with limited computational resources.

TABLE V. COMPARISON BEFORE AND AFTER IMPROVEMENT

Algorithm	Pars/M	FLOPS/G	FPS/s/f	mAP/%
RT-DETR	19.8	57.3	69	67.1
Yolov11m	20.1	68.0	107	67.9
Fast-RCNN	136.5	370.2	21	50.2
Improved RT-DETR	14.6	45.2	60	69.2

V. COPYRIGHT FORMS AND REPRINT ORDERS

This paper addresses the issues of high false detection rates in complex road damage detection scenarios and missed detection of potholes by improving the RT-DETR network model. We propose an efficient backbone network for long-range semantic feature extraction to reduce computational overhead and mitigate false detections in complex environments. Additionally, a feature pyramid network incorporating Full Kernel modules and SPDConv is introduced to small-target enhanced feature pyramid network, specifically addressing the problem of missing tiny potholes. A series of experiments have demonstrated the effectiveness of the proposed algorithm. While the improved model shows enhanced detection performance, there remains room for optimization as it still exhibits relatively high computational complexity and parameter volume, along with decreased FPS compared to the original RT-DETR. Future work will focus on optimizing the model scale and improving detection speed.

REFERENCES

[1] Hao S, Shao L, Wang S. A Faster RCNN Airport Pavement Crack Detection Method Based on Attention Mechanism [J]. Academic Journal of Science and Technology, 2022, 4(2): 129-132.

[2] Redmon J, and Farhadi A. YOLOv3: An Incremental Improvement [J]. CoRR, 2018, 1804: 02767.

[3] Wu L, Duan Z, Liang C. Research on asphalt pavement disease detection based on improved YOLOv5s[J]. Journal of Sensors, 2023, 2023(1): 2069044.

[4] DOSOVITSKIY A, BEYER L, KOLESNIKOV A, et al. An image is worth 16x16 words: Transformers for image recognition at scale [J]. arXiv preprint arXiv:2010.11929, 2020.

[5] CARIONN, MASSAF, SYNNAEVE G, et al. End-to-end object detection with transformers[C]// Proceedings of the 2020 European Conference on Computer Vision. Cham: Springer International Publishing, 2020: 213-229.

[6] Zhu X, Su W, Lu L, et al. Deformable detr: Deformable transformers for end-to-end object detection[J/OL]. arXiv preprint arXiv, 2020[2024-11-18]. https://doi.org/10.48550/arXiv.2010.04159

[7] CHEN Q, CHENX, WANGJ, et al. Group detr: Fast detr training with group-wise one-to-many assignment [C]//Proceedings of the 2023 IEEE/CVF International Conference on Computer Vision. Piscataway: IEEE, 2023: 6633-6642.

[8] Zhang H, Li F, Liu S, et al. Dino: Detr with improved denoising anchor boxes for end-to-end object detection [J]. arXiv preprint arXiv:2203.03605, 2022.

[9] Zong Z, Song G, Liu Y. Detrs with collaborative hybrid assignments training [C]//Proceedings of the IEEE/CVF international conference on computer vision. 2023: 6748-6758.

[10] Chen Y, Zhang C, Chen B, et al. Accurate leukocyte detection based on deformable-DETR and multi-level feature fusion for aiding diagnosis of blood diseases[J]. Computers in Biology and Medicine, 2024, 170: 107917.

[11] Zhao Y, Lv W, Xu S, et al. Detrs beat yolos on real-time object detection [C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024: 16965-16974.

[12] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition [C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[13] Wang C Y, Yeh I H, Mark Liao H Y. Yolov9: Learning what you want to learn using programmable gradient information [C]//European conference on computer vision. Cham: Springer Nature Switzerland, 2024: 1-21.

[14] Ding X, Zhang Y, Ge Y, et al. UniRepLKNet: A Universal Perception Large-Kernel ConvNet for Audio Video Point Cloud Time-Series and Image Recognition [C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2024: 5513-5524.

[15] Sunkara R, Luo T. No more strided convolutions or pooling: A new CNN building block for low-resolution images and small objects [C]//Joint European conference on machine learning and knowledge

discovery in databases. Cham: Springer Nature Switzerland, 2022: 443-459.

[16] Cui Y, Ren W, Knoll A. Omni-Kernel Network for Image Restoration [C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2024, 38(2): 1426-1434.

[17] Arya D, Maeda H, Ghosh S K, et al. RDD2020: An annotated image dataset for automatic road damage detection using deep learning [J]. Data in brief, 2021, 36: 107133.

[18] Everingham M, Van Gool L, Williams C K I, et al. The pascal visual object classes (voc) challenge[J]. International journal of computer vision, 2010, 88: 303-338.

[19] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards real-time object detection with region proposal networks [J]. IEEE transactions on pattern analysis and machine intelligence, 2016, 39(6): 1137-1149.

[20] Khanam R, Hussain M. Yolov11: An overview of the key architectural enhancements [J]. arXiv preprint arXiv:2410.17725, 2024.

Research on Vehicle and Pedestrian Detection Based on Improved RT-DETR

Jingshu LI
School of Computer Science and Engineering
Xi'an Technological University
Xi'an, 710021, China
E-mail: lijs812@163.com

Jianguo Wang
State and Provincial Joint Engineering Lab. of Advanced Network, Monitoring and Control
Xi'an Technological University
Xi'an, 710021, China
E-mail: wjg_xit@126.com

***Abstract*—This paper proposes a vehicle and pedestrian detection model based on an improved RT-DETR to address the issues of high redundancy in feature extraction and insufficient accuracy for small targets in existing real-time detection models, especially in complicated traffic scenarios. The core of this improved model is to embed a parameter free SimAM (Simple Attention Module) attention mechanism in the backbone network. The SimAM mechanism dynamically generates three-dimensional attention weights through energy functions, effectively enhancing the expression ability of fine-grained features of pedestrians and vehicles. This improvement not only reduces redundant information in the feature extraction process, but also improves the detection accuracy of the model for small targets, enabling the model to more accurately identify and locate small targets when dealing with complex traffic scenes. The experimental results show that on the BDD100K dataset, the improved model achieved an average precision of 73.6%, which is 3.7 percentage points higher than the original RT-DETR, effectively enhancing the model's capability to detect vehicles and pedestrians in intricate environments.**

Keywords-Object Detection; RT-DETR; Attention Mechanism; Autonomous Driving

I. INTRODUCTION

Today, the technology for detecting vehicles and pedestrians stands as a key component in multiple domains. Especially in the realm of autonomous driving, precisely and swiftly recognizing vehicles and pedestrians is fundamental to guaranteeing the safety and reliability of self-driving cars. However, real-world traffic scenarios have brought many challenges to detection technology. For example, frequent occurrences of mutual occlusion between vehicles and partial occlusion of pedestrians by roadside obstacles make detection algorithms prone to missed or false detections. In addition, under low light conditions, such as night or rainstorm weather, the image clarity and contrast will be significantly reduced, which undoubtedly brings great challenges to the detection task.

The fast-paced growth of deep learning has catalyzed substantial advancements in vehicle and pedestrian detection methods. Early detection techniques were largely facilitated by manually designed characteristics and conventional machine learning algorithms like SIFT, HOG, etc. However, the effectiveness of these methods was not ideal and there were many limitations. Subsequently, deep learning based object detection methods gradually gained prominence, which are broadly classified into single-stage object detectors and two-stage object detectors. Single stage object detectors, such as YOLO series [1], SSD [2], RetinaNet [3], directly forecast the location and type of the target on the input image without the need for complex candidate region generation steps, thus having high detection accuracy. Two stage object detectors like Fast R-CNN [4], Cascade R-CNN, CBNet, etc., they begin by creating potential regions, followed by classification and bounding box regression for each one. Although the detection accuracy is high, the process tends to be slow. In these detection methods, a large number of anchor boxes are generated during the detection phase, however, for an object, only one detection box is actually needed to represent it. Therefore, it is necessary to discard overlapping detection boxes through Non Maximum Suppression (NMS) methods to guarantee that each target is identified by a single box. In addition, intricate parameter tuning is

necessary during network training to optimize detection performance.

In view of this, researchers have shifted their focus to the Transformer structure, which has shown outstanding performance in the field of natural language processing, hoping to bring new breakthroughs to computer vision with its powerful feature extraction and modeling capabilities. As a result, a series of new structures based on Transformers emerged, such as Vision Transformer (VIT) [5] and Detection Transformer (DETR). However, although the DETR series models based on Transformer adopt a non-maximum suppression (NMS) architecture, which solves the problem of slow inference speed caused by traditional object detection models relying on NMS, their high computational cost cannot meet real-time detection requirements, and they have not shown significant advantages in inference speed. This issue limits the widespread adoption of DETR series models [6] in practical applications, especially in scenarios that require high real-time performance. To solve this problem, researchers have continuously improved and optimized the model, resulting in many excellent variants such as Deformable DETR, Conditional DETR, DAB-DETR (Dynamic Anchor Boxes Are Better Questions for DETR), RT-DERT, etc. These variants have improved the model's efficiency and performance through various novelty strategies, upholding the strengths of the Transformer architecture, making it closer to the requirements of practical applications [7].

Among numerous improved DETR models, RT-DETR has received widespread attention for its excellent real-time performance and detection accuracy. However, RT-DETR still has some shortcomings in vehicle and pedestrian detection tasks. For example, RT-DETR may experience a decrease in detection accuracy due to background interference when dealing with complex scenes. In addition, the detection performance of RT-DETR needs to be improved for small and occluded targets.

In response to these issues, this article chooses RT-DETR as the baseline model for improvement. By replacing some HGBlock modules in the backbone network with HGBlock_SimAM modules, the model can focus on important information in the image earlier, thereby preventing introducing too many extraneous or duplicate features in the initial stage. This advancement raises the model's detection accuracy and also enhances its operational performance to a notable degree.

II. Related Work

A. Application and Improvement of RT-DETR in Vehicle Inspection

RT-DETR, as an efficient real-time object detection model, has demonstrated significant performance in vehicle detection tasks. However, there is still room for improvement in detection accuracy in complex traffic backgrounds, especially when dealing with small targets and background interference. To overcome these challenges, researchers have proposed various improvement strategies. For example, Azimjonov [8] proposed a vehicle detection algorithm based on improved RT-DETR, which significantly improves the model's detection ability for small targets by introducing multi-scale feature fusion and global information. In addition, Ghosh [9] suggested an approach utilizing Faster R-CNN for vehicle detection under different weather conditions, improving accuracy via the enhancement of the Region Proposal Network (RPN). These studies provide new ideas for the application of RT-DETR in vehicle detection.

B. Application and Improvement of RT-DETR in Pedestrian Detection

Spotting pedestrians is a key responsibility in computer vision, regularly used in applications like autonomous driving systems and video security. Although RT-DETR performs well in pedestrian detection, there are still some shortcomings when dealing with occlusions, complex backgrounds, and small targets. In order to improve the performance of RT-DETR in pedestrian detection, researchers have made multiple improvements. For example, Ma [10] et al. presented a fuzzy-logic enhanced pedestrian detection strategy using DETR, which significantly improved the model's detection accuracy for occluded pedestrians by introducing dynamic deformable convolution and cascaded Transformer decoders. In addition, Xing [11] et al. proposed a multispectral pedestrian detection Transformer (MS-DETR), which further enhances the detection capability of the model in complex

environments by fusing visible light and thermal imaging features.

C. Multi-scale Fusion and Small Target Enhancement of RT-DETR

In order to further improve the performance of RT-DETR in vehicle pedestrian detection, researchers have also proposed improved methods such as multi-scale feature fusion and small target enhancement. For example, Wei [12] et al. proposed an improved model RT-DETR-MSS based on RT-DETR, which significantly improves the model's detection ability for small targets by introducing a Multi Scale Fusion Module and a Small Object Enhancement Structure. In addition, the study also introduced GSConv and Slim Neck structures to optimize the network structure and improve computational efficiency. The experimental results indicate that RT-DETR-MSS performs well on CrowdHuman and WiderPerson datasets mAP@50 Increased by 1.7% and 0.8% respectively, mAP@50:95 increased by 2.2% and 1.2% respectively.

D. Real Time and Accuracy Optimization of RT-DETR

In practical applications, the real-time capabilities and precision in detection of RT-DETR are crucial. In order to meet real-time requirements, researchers have further improved the efficiency of RT-DETR by optimizing the model structure and training strategy. For example, Sadik [13] et al. proposed a deep learning framework construct using YOLOv8 and RT-DETR for the immediate recognition of vehicles and pedestrians. This study conducted experiments within complex urban environments, and the results showed that the YOLOv8 Large version performs well in identifying pedestrians, offering high precision and reliability. In addition, the study emphasizes the importance of maintaining high accuracy and reliability under different environmental conditions, such as crowded streets, changing weather, and different lighting scenarios.

In summary, RT-DETR has demonstrated significant performance in vehicle and pedestrian detection tasks, but there are still some challenges such as small object detection, background interference, and real-time performance. To overcome these challenges, researchers have proposed various improvement strategies, including multi-scale feature fusion, small target enhancement, dynamic deformable convolution, and cascaded Transformer decoders. These improvements significantly enhance the detection precision and efficiency of RT-DETR in complex scenarios, providing strong support for applications in the fields of autonomous driving and video surveillance.

III. TECHNICAL MODEL

A. RT-DETR Model

RT-DETR [14] is an efficient real-time object detection model with a well-designed architecture consisting of three key components, a backbone network, an efficient hybrid encoder, and a Transformer decoder that includes a supplementary prediction head. Specifically, the backbone network of RT-DETR is responsible for extracting multi-scale features of images. In this model, the features of the last three stages (S3, S4, S5) of the backbone network are sent to the encoder. An efficient hybrid encoder is one of the core components of RT-DETR, which transforms multi-scale features into a series of image features with rich semantic information through intra scale feature interaction (AIFI) and cross scale feature fusion (CCFM). This feature fusion method can effectively capture detailed information and overall contextual data within images, providing robust feature representation for subsequent object detection. Subsequently, RT-DETR adopts an uncertainty minimization query strategy, selecting a fixed number of features from the encoder output as the initial object query. These initial queries are then refined through iterative optimization in a decoder with auxiliary prediction heads, ultimately generating the target category and bounding box. The overall architecture of the RT-DETR model is shown in Figure 1, which clearly illustrates the various components of RT-DETR and the data flow between them.

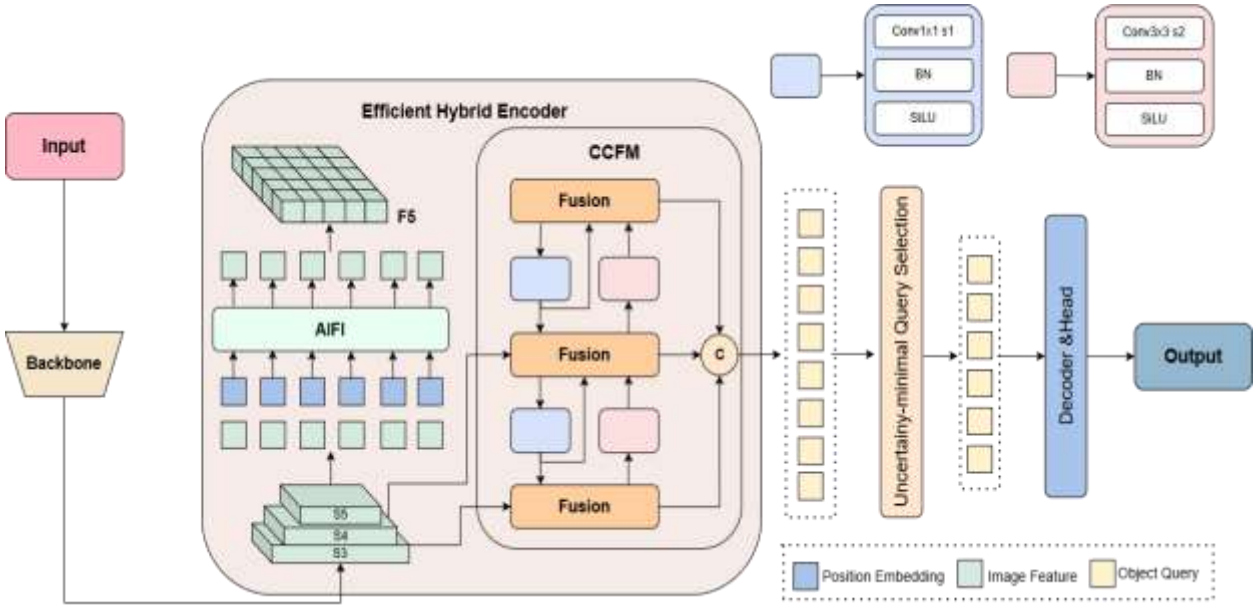


Figure 1. Network architecture of RT-DETR

B. Improved RT-DERT

In the original RT-DETR model, although it performs well in real-time, there are still some limitations. For example, when dealing with complex scenes, models may introduce too much irrelevant or redundant feature information, which not only reduces the detection accuracy of the model, but also elevates the computational workload. To overcome these potential obstacles, this paper recommends substituting some HGBlock modules in the backbone network with HGBlock_SimAM modules. By integrating SimAM attention mechanism in the HGBlock module [15], the model can focus on important information in the image earlier, thereby avoiding introducing too many irrelevant or redundant features in the initial stage. The enhancement not only boosts the model's detection precision but also enhances its operational efficiency to a certain degree. The improved RT-DETR structure is shown in Figure 2. Through experimental verification, the improved RT-DETR model has significantly improved detection accuracy in complex scenes compared to the original model.

Figure 2. Network architecture of improved RT-DETR

C. Principle of SimAM (Simple Attention Module) Attention Mechanism

SimAM is a simple and parameter free attention mechanism designed to provide an efficient and parameter free attention module for neural networks. It is based on the spatial inhibition theory

in neuroscience, using optimization of specific energy functions to measure the importance of each neuron. Specifically, SimAM is implemented through the following steps.

1) Energy function optimization

The aim in optimizing the energy function is to identify the most suitable weights for each neuron, which will represent their importance within the feature map. The energy function defined by SimAM is as follows.

$$E(t)=\sum_{i\in N}\left(\mathrm{y_i}-\hat{\mathrm{t}}\right)^2+\lambda\sum_{i\in N}\sum_{\mathrm{j}\in N}\left(\mathrm{y_i}-\mathrm{y_j}\right)^2 \quad (1)$$

Among them, t is the target neuron, N is the domain of the target neuron, y_i is the activation value of the i-th neuron, $\hat{t}$ is the predicted value of the target neuron,and λ is the regularization parameter used to balance the weights of the two terms.

2) Quick analytical solution

SimAM proposed a fast analytical solution for efficiently calculating the weights of each neuron. The analytical solution formula is as follows.

$$w_i=\frac{1}{k}\sum_{j\in N_i}s(f_i,f_j) \quad (2)$$

$$s(f_i,f_j)=-\|f_i-f_j\|_2^2 \quad (3)$$

Among them, w_i is the attention weight of the i-th neuron, k is the normalization constant, N_i is the domain of the i-th neuron, $s\left(f_i,f_j\right)$ is the similarity between the i-th neuron and the j-th neuron.

3) Attention weight calculation

Finally, SimAM calculates the attention weight of each neuron using the following formula.

$$w_i=\frac{1}{e^*} \quad (4)$$

Among them, e^* is achieved by optimizing the energy function. The greater the attention weight w_i, the more significant the role of the neuron i.

SimAM differs from traditional channel attention mechanisms and spatial attention modules in that it can directly infer the three-dimensional attention weights of feature maps without increasing the original network parameters, referencing Figure 3. In vehicle and pedestrian detection tasks, SimAM helps the model better focus on the target area, thereby improving detection accuracy.

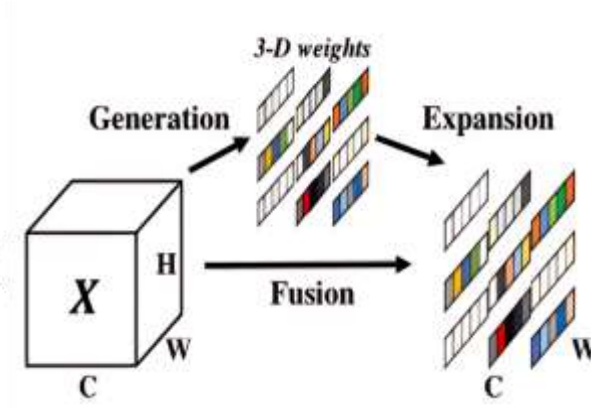


Figure 3. Full 3-D weights for attention

IV. Experiment and Analysis

A. Experimental environment

The experiments in this article were conducted on servers provided by AutoDL. The parameters configured for the experimental conditions are shown below, the system operates on Ubuntu 22.04 and is powered by 24 virtual CPU cores of the Intel (R) Xeon (R) Platinum 8255C model, running at a 2.5GHz clock speed, complemented by two RTX 3080 GPUs, each with 10GB of storage. In terms of software framework, PyTorch version 2.3.0, Python version 3.12, and CUDA version 12.1.0 are used. The explicit experimental training parameters are depicted in Table 1.

TABLE I. EXPERIMENTAL PLATFORM

Hyper-parameters	Value
Inputs	640x640
Epochs	100
Batchsize	16
Lr0	0.001
Lrf	0.0001
Momentum	0.9
Warmup-decay	0.0005
Warmup-epochs	5

B. Experimental environment

In order to evaluate the performance of improving RT-DERT detection of vehicles and pedestrians, the dataset used in this paper is BDD100K (Berkeley DeepDrive 100k). BDD100K is a large-scale and diverse dataset designed specifically for autonomous driving research, containing 100000 driving scene images covering complex scenarios such as different weather conditions (sunny/rainy/snowy), lighting conditions (day/dusk/night), and geographic regions (urban and rural roads in the United States/Asia). This dataset can be used to specifically test the performance of our detection system in scenarios with dense vehicular and pedestrian traffic. Table I presents detailed information on dataset sampling and sample partitioning in this study, covering key aspects such as dataset size, sampling method, sample size, and sample partitioning ratio.

In order to further improve the generalization ability of the model under small sample conditions, this study adopts transfer learning strategy and uses a pre trained model on the COCO2017 dataset. COCO2017 is a widely used dataset that contains rich categories and complex scenes, and its pre trained models can provide a good initialization weight. Following that, perform adjust the pre-trained model utilizing the 3000 extracted instances. Experimental results have shown that compared to training models directly from scratch on small sample datasets, this strategy of transfer learning combined with fine-tuning significantly improves the mean Average Precision (mAP@50) on the validation set during the initial training phase.

TABLE II. DATASER SAMPLING SITUATION

Content	Detailed information
Dataset size	69534 valid training samples
Sample method	Randomly select samples
Sample quantity	Sampling 3000 samples
Tag filtering	Filter other category tags
Division ratio	8:2
Training	2400 training images
Verify	600 verification images

C. Evaluation

This experiment uses Precision, Recall mAP@50 and mAP@50:95 measures to assess the model's effectiveness, with comprehensive descriptions of these criteria outlined below.

1) Precision reflects the model's precision in identifying positive samples by showing the percentage of correct positive predictions. The calculation formula is as follows.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

Among them, TP is the abbreviation for true positive, which is the count of positive samples that were accurately predicted, and FP is for false positive, which is the count of positive samples that were inaccurately predicted.

2) Recall measures the proportion of samples that are actually positive and correctly predicted as positive by the model. The calculation formula is as follows.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

Among them, FN, meaning false negative, is used to describe the number of negative samples that were misidentified.

3) MAP (Mean Average Precision) determines the model's average performance across all categories by averaging the AP (Average Precision) values for each one. The calculation formulas are as follows.

$$AP = \int_0^1 Precision\left(Recall\right) d\left(Recall\right) \tag{7}$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} \mathrm{AP}_i \quad (8)$$

Among them, N is the total number of categories and is the AP value of the i-th category.

mAP@50 is the value of mAP when the IoU (Intersection over Union) threshold is 0.50. In object detection tasks, IoU is used to measure the degree of overlap between the predicted bounding box and the ground-truth bounding box. mAP@50 calculates the average of the AP (Average Precision) values of all categories at IoU=0.50.

mAP@50:95 is the average value of mAP when the IoU threshold ranges from 0.50 to 0.95. The specific calculation method is to calculate the AP values for all categories for each IoU value (from 0.50 to 0.95, usually in steps of 0.05), and then take the average of these AP values to obtain the mAP value. Finally, the mAP values corresponding to all IoU values are averaged to obtain mAP@50:95.

D. Experimental results

Extensive experiments were conducted to validate the effectiveness of adding SimAM attention mechanism in the RT-DETR backbone network. The experimental results indicate that this enhancement strategy augments the model's performance.

1) Improved detection accuracy

The changes in evaluation metrics of the RT-DETR model under different configurations are depicted in Figure 4. The left chart shows the situation of the RT-DETR model without SimAM attention mechanism. The accuracy (Rrecall) oscillates frequently in the range of 0.72 to 0.78, and although the recall remains above 0.7, there is a periodic decline of 20%. mAP@50 The final convergence is around 0.65, and the growth is weak, while the sum of strict detection ability is measured mAP@50:95 has remained stagnant below the 0.35 threshold for a long time. In contrast, after using the HGBlock_SimAM module on the right side, the curves of various indicators are smoother, including accuracy, recall, mAP50, and mAP@50:95 steadily improved during the training process, mAP@50 Breaking through 0.7, this demonstrates the positive effect of introducing SimAM attention mechanism on model performance optimization.

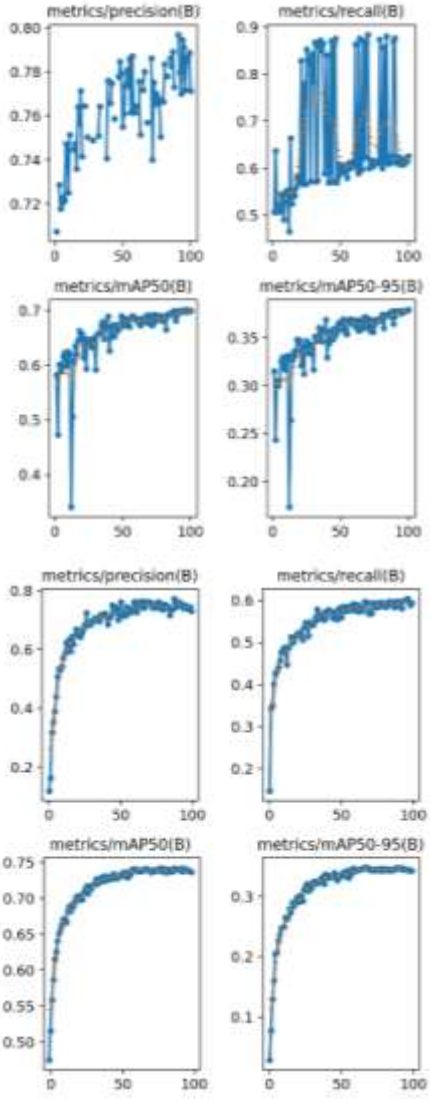


Figure 4. Comparison before and after improvement

Specifically, mAP@50 The RT-DETR has increased from 0.699 to 0.736, an increase of 3.7 percentage points. This result indicates that the SimAM attention mechanism allows the model to concentrate on crucial information in the image earlier, thereby detecting target objects more accurately. The specific experimental outcomes are depicted in Table 3.

TABLE III. EXPERIMENTAL RESULTS

RT-DETR	Without SimAM	Added SimAM
Precision	0.779	0.793
Recall	0.621	0.624
mAP@50	0.699	0.736
mAP@50:59	0.379	0.383

2) Training and reasoning efficiency

The implementation of the SimAM attention mechanism has not led to a significant decrease in the model's training and inference efficiency. As training progresses, the model's rate of convergence remains steady, and the training duration is equivalent to the original RT-DETR's. In the inference step, the advanced RT-DETR accomplishes real-time object detection, with a minor decline in inference speed relative to the original model, indicating that the SimAM mechanism improves performance while maintaining the efficiency of the model.

3) Comparison with other models

For validation of the proposed method's overall efficacy, a comparison and analysis were made with the traditional object detection algorithm YOLOv8. Experiments were executed on the BDD dataset, and findings are detailed in Table 4.

According to Table 4, the model proposed in this paper is more effective than YOLOv8. In detail, the accuracy has jumped from 0.721 with YOLOv8 to 0.793, mAP@50 has also increased from 0.692 to 0.736, and mAP@5095 has escalated from 0.372 to 0.383. The findings demonstrate that incorporating the SimAM attention mechanism has successfully enhanced the model's detection precision and its ability to perform fine-grained detections.

TABLE IV. CAMPARISON RESULTS

Model	YOLOv8	Ours
Precision	0.721	0.793
Recall	0.645	0.624
mAP@50	0.692	0.736
mAP@50:59	0.372	0.383

4) Visualization results

The actual detection results in different scenarios of the BDD dataset test set are shown in Figure 5.

Figure 5. Visualization results

The improved RT-DETR model proposed in this study demonstrates significant advantages in complex scenarios. Specifically, when dealing with partial occlusion, the model secured an 80% accuracy in identifying the locations of both vehicles and pedestrians. This indicates that the improved model can more accurately identify and locate targets when dealing with complex scenes and occlusion problems, thereby significantly improving detection performance.

V. CONCLUSIONS

This article proposes an improved model based on RT-DETR, which achieves an average accuracy value (mAP) of 73.6% on the BDD dataset by replacing some HGBlock modules in the backbone network with HGBlock_SimAM modules. This result is superior to the original RT-DETR model, fully demonstrating the effectiveness of introducing SimAM attention mechanism in the field of vehicle and pedestrian detection.

Moving forward, we intend to keep refining and enhancing the algorithm of this model. On the one hand, the plan is to further reduce the number of parameters in the model to enhance its computational speed and real-time capabilities, making it more suitable for use in resource constrained environments, such as real-time detection systems for embedded devices or autonomous vehicles. On the other hand, efforts

will be geared towards enhancing the model's detection precision by introducing more advanced feature extraction and fusion techniques to further strengthen its detection capabilities for small targets, occluded targets, and complex backgrounds. In addition, the scalability and adaptability of the model will be explored to better cope with changes in different scenarios and datasets. Through these optimization and improvement measures, the improved RT-DETR model is expected to play a greater role in the field of autonomous driving detection, providing strong technical support and reference for the development of autonomous driving technology.

REFERENCES

[1] Hidayatullah, P.; Syakrani, N.; Sholahuddin, M. R.; Gelar, T.; Tubagus, R. YOLOv8 to YOLOv11: A Comprehensive Architecture In-Depth Comparative Review.

[2] Zhang, X.; Zhang, Y.; Gao, T.; Fang, Y.; Chen, T. A Novel SSD-Based Detection Algorithm Suitable for Small Object. IEICE Trans. Inf. Syst. 2023, E106.D (5), 625–634.

[3] Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal Loss for Dense Object Detection. IEEE Trans. Pattern Anal. Mach. Intell. 2020, 42 (2), 318–327.

[4] Arora, N.; Kumar, Y.; Karkra, R.; Kumar, M. Automatic Vehicle Detection System in Different Environment Conditions Using Fast R-CNN. Multimed. Tools Appl. 2022, 81 (13), 18715–18735.

[5] Abd Alaziz, H. M.; Elmannai, H.; Saleh, H.; Hadjouni, M.; Anter, A. M.; Koura, A.; Kayed, M. Enhancing Fashion Classification with Vision Transformer (ViT) and Developing Recommendation Fashion Systems Using DINOVA2. Electronics 2023, 12 (20), 4263.

[6] Fahad, I. A.; Arean, A. I. H.; Ahmed, N. S.; Hasan, M. Automatic Vehicle Detection Using DETR: A Transformer-Based Approach for Navigating Treacherous Roads. arXiv February 25, 2025.

[7] Cheng Xinmiao, Zhang Xuesong, Cao Bingjie, Song Cunli Research on Improving the Small Object Detection Method of RT-DETR [J]. Computer Engineering and Applications, 1-21.

[8] Azimjonov, J.; Özmen, A. A Real-Time Vehicle Detection and a Novel Vehicle Tracking Systems for Estimating and Monitoring Traffic Flow on Highways. Adv. Eng. Inform. 2021, 50, 101393.

[9] Ghosh, R. On-Road Vehicle Detection in Varying Weather Conditions Using Faster R-CNN with Several Region Proposal Networks. Multimed. Tools Appl. 2021, 80 (17), 25985–25999.

[10] Wu, T.; Li, X.; Dong, Q. An Improved Transformer-Based Model for Urban Pedestrian Detection. Int. J. Comput. Intell. Syst. 2025, 18 (1), 68.

[11] Xing, Y.; Yang, S.; Wang, S.; Zhang, S.; Liang, G.; Zhang, X.; Zhang, Y. MS-DETR: Multispectral Pedestrian Detection Transformer with Loosely Coupled Fusion and Modality-Balanced Optimization. IEEE Trans. Intell. Transp. Syst. 2024, 25 (12), 20628–20642.

[12] Song, Y.; Qian, P.; Zhang, K.; Liu, S.; Zhai, R.; Song, R. An Improved Multi-Scale Fusion and Small Object Enhancement Method for Efficient Pedestrian Detection in Dense Scenes. Multimed. Syst. 2025, 31 (2), 151.

[13] Sadik, M. N.; Hossain, T.; Sayeed, F. Real-Time Detection and Analysis of Vehicles and Pedestrians Using Deep Learning. arXiv April 11, 2024.

[14] Zhao, Y.; Lv, W.; Xu, S.; Wei, J.; Wang, G.; Dang, Q.; Liu, Y.; Chen, J. DETRs Beat YOLOs on Real-Time Object Detection. arXiv April 3, 2024.

[15] Xu, Y.; Du, W.; Deng, L.; Zhang, Y.; Wen, W. Ship Target Detection in SAR Images Based on SimAM Attention YOLOv8. IET Commun. 2024, 18 (19), 1428–1436.

A Course Recommendation Method Based on the Integration of Curriculum Knowledge Graph and Collaborative Filtering

Jingyi Hu
Big Data and Artificial Intelligence College
Anhui Xinhua University
Hefei, China
E-mail: hujingyi@axhu.edu.cn

Qingqing Wang
Big Data and Artificial Intelligence College
Anhui Xinhua University
Hefei, China
E-mail: 3119905948@qq.com

***Abstract*—To address the problems of data sparsity and cold start in collaborative filtering algorithms, this paper proposes an improved course recommendation method that integrates knowledge graphs and collaborative filtering. First, the RippleNet model is used to construct a knowledge graph based on course-attribute-relation triples and generate a recommendation list. Then, an item-based collaborative filtering algorithm utilizes users' historical interaction behavior to produce another recommendation list. Finally, a weighted linear method is employed to fuse the recommendation list generated by the RippleNet-based course knowledge graph and the one generated by collaborative filtering, resulting in the final course recommendation list. Experiments conducted on the public dataset MOOCCube demonstrate that the RippleNet-CF method improves precision, recall, and F1-score, while also effectively mitigating the issue of data sparsity.**

Keywords-Data Sparsity; Course Attributes; Knowledge Graph

I. INTRODUCTION

With the rapid development of information technology, there has been an explosion of data [1], and data mining technology has been widely applied in various fields such as education, communication and e-commerce [2]. In the field of education, how to recommend courses based on students' learning characteristics is a key focus of data mining [3]. In recommendation systems, domain-based recommendation is the most fundamental algorithm, which is generally divided into user-based collaborative filtering algorithms [4] and item-based collaborative filtering algorithms [5]. The user-based collaborative filtering algorithm recommends items to target users based on user similarity. When the target user has too few historical interactions with items, it cannot make accurate recommendations. This algorithm is more suitable for social recommendations such as news [6]. This algorithm is suitable for personalized user recommendations but also has the problem that too few interactions between users can lead to unsatisfactory recommendation results. In order to optimize the recommendation effect, scholars have considered introducing and expanding the sources of information. They can use auxiliary information such as the attributes of items themselves, users' social networks, and context to improve the accuracy of recommendations.

This paper proposes a RippleNet-CF model that combines the RippleNet model based on knowledge graphs and the collaborative filtering algorithm. The algorithm leverages course entities and the attributes of courses themselves to simulate the propagation of user course interests on the knowledge graph through ripple patterns. It also takes into account the interaction history between users and courses, such as viewing records and ratings, to uncover personalized recommendations for users. By expanding the sources of information and integrating the historical and current interests of target users, the accuracy of recommendation results is enhanced. The performance of the recommendation results is evaluated using three metrics: accuracy, recall, and F1.

DOI: 10.2478/ijanmc-2025-0020

II. Related Theories

A. Collaborative Filtering Recommendation Algorithm

The item-based collaborative filtering algorithm calculates the similarity between courses based on user preference data and then recommends a list of other courses that are similar to the ones the user likes [7]. However, it faces issues such as data sparsity and cold start. This paper chooses the course-based collaborative filtering algorithm for personalized course recommendations, and the implementation of this algorithm is divided into two steps:

1) Calculate the similarity between courses

Construct a student-course matrix:Let $U=\{u_1,u_2,u_3,..,u_m\}$ be the set of m students; $I=\{i_1,i_2,i_3,\cdots,i_n\}$ be the set of n courses, and Rm $\times$ n represent the rating matrix of students to courses as shown in formula (1):

$$R_{m\times n}=\begin{bmatrix} R_{11} & R_{12} & R_{13} & \cdots & R_{1n-1} & R_{1n} \\ R_{21} & R_{22} & R_{23} & \cdots & R_{2n-1} & R_{2n} \\ R_{31} & R_{32} & R_{33} & \cdots & R_{3n-1} & R_{3n} \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ R_{m-11} & R_{m-12} & R_{m-13} & & R_{m-1n-1} & R_{m-1n} \\ R_{m1} & R_{m2} & R_{m3} & & R_{mn-1} & R_{mn} \end{bmatrix} \quad (1)$$

Here, R_{ij} represents the rating of student U_i to course I_j, and the higher the value of R_{ij}, the more student u_i likes course I_j.

As an example, to measure how similar two courses are, all students' ratings for a given course are treated as an m $\times$ 1 vector. The ratings for course i are represented as $F_i = \{r_{1i}, r_{2i}, r_{3i}, ..., r_{mi}\}$, and the ratings for course j are recorded as $F_j = \{r_{1j}, r_{2j}, r_{3j}, ..., r_{mj}\}$. The formula for computing the similarity between courses i and j is provided in Equation (2).

$$W_{ij}=\frac{F_i\cdot F_j}{\|F_i\|\cdot\|F_j\|}=\frac{\sum_{u=1}^{m} r_{u_i}\cdot r_{uj}}{\sqrt{\sum_{u=1}^{m} r_{u_i}{}^2}\cdot\sqrt{\sum_{u=1}^{m} r_{u_j}{}^2}} \quad (2)$$

Among them, W_{ij} represents the cosine similarity value between course i and course j, with a corresponding range of [-1, 1]. The W_{ij} higher the value, the more similar courses i and j are, and the target user is expected to have similar behavior towards the course in the future.

2) Selecting Neighbors

When selecting neighbors, this paper chooses to rank them according to the similarity of courses. Then, several courses with the highest ranks from the sorted results are selected as neighbors.

B. Knowledge Graph Learning

Knowledge Graphs (KG) [8] can effectively map out vast amounts of disordered data through theoretical methods such as data mining and information processing, making it more convenient and accurate for people to obtain the information they need. A knowledge graph is a large-scale semantic network representing a complex web of relationships between entities, generally composed of (entity, relationship, entity) triples [9]. Incorporating knowledge graphs into recommendation systems can uncover deeper semantic relationships and more precisely identify the interests of target users. Currently, the application of knowledge graph feature learning [10] in recommendation systems is generally divided into: path-based recommendation algorithms [11]and embedding-based recommendation algorithms [12], with representative models including TransE, TransH, SME, NTN, etc.

C. RippleNet Model

Due to the limitations of knowledge graph perception reconstruction methods applied to recommendation systems, scholars have proposed another model, RippleNet [13].

The knowledge graph is constructed from the triple relationships corresponding to course entities $G=\{(h,r,t)\,|\,h,t\in R\}$. The goal of the RippleNet model is to construct a knowledge graph to utilize students' preferences for courses and calculate the click probability of student u for the target course v. The main implementation of its algorithm is as follows:

Deffnition 1: Item Embedding. Based on the characteristics, semantics, and attributes of items, the embedding is performed. Given the embedding vector v of a speciffed course and the 1-hop ripple

set, each expansion outward yields a triplet. The relevance score between item v and each (h_i, r_i, t_i) in the 1-hop set is calculated, and the linear relevance scores are normalized using the softmax function. Consequently, the head entity h_i and the relation R_i of the triplet are treated as an association probability P_i, as shown in formula (3):

$$p_i = soft\max(v^T R_i h_i) = \frac{\exp\left(v^T R_i h_i\right)}{\sum_{(h,r,t)\in S_u^1} \exp(v^T Rh)} \quad (3)$$

$$o_u^1 = \sum_{(h_i,r_i,t_i)\in S_u^1} p_i t_i \quad (4)$$

Here, v^T represents the item vector, t_i is the tail entity vector, h_i is the head entity vector, and r_i is the relation mapping matrix. S_u^1 the first-layer Ripple Set of a student (the first-hop Ripple Set, as shown in the figure1) is formed by selecting a certain number of items from the student ' s interaction history.Essentially, this process calculates the correlation and similarity between the seed node and its connected one-hop nodes in the knowledge graph, as represented by triples—illustrated in Figure 1.

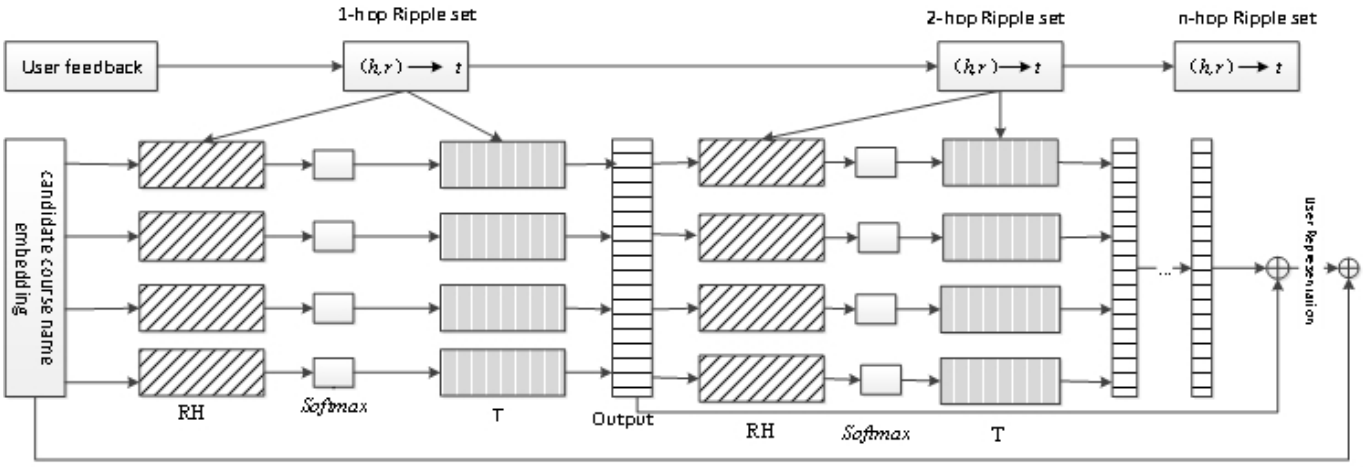


Figure 1. RippleNet Model Diagram.

By repeating the above process, the knowledge graph undergoes multi-hop propagation. The corresponding vectors obtained from each hop are then summed to generate the student's embedding vector (user embedding). After repeating the process H times, H output vectors o are obtained, and the final user embedding is calculated according to Equation (5).

$$u = o_u^1 + o_u^2 + \ldots + o_u^H \quad (5)$$

Finally, the likelihood of user u engaging with course v is computed by integrating their respective latent representations, as illustrated in Equation (6).

$$y_{uv} = \sigma(u^T v)$$
$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (6)$$

III. Integration of the RippleNet Model and an Item-Oriented Collaborative Filtering Approach

Conventional item-level recommendation techniques algorithms only consider users' rating data on courses. After extracting the relationships between courses and their attributes, this paper proposes an algorithm that integrates course attribute information with user-course interaction data by combining the RippleNet model and collaborative filtering. The RippleNet model

leverages historical user-course rating records as implicit relationships between users and items. It constructs a knowledge graph based on the relationships among course attributes and extracts corresponding triples for each course. Using the ripple propagation mechanism through these triples, it computes user preferences. Meanwhile, the item-oriented filtering method estimates a user's interest in unvisited courses by analyzing past interactions between the user and various courses. By combining both approaches, a comprehensive course recommendation list is generated. This method fully utilizes the strengths of both algorithms by linearly fusing the results of the two recommendation lists. The fusion method is defined in Equation (7).

$$C = \beta * Y + (1 - \beta) * P \tag{7}$$

Here, β represents the weight within the range (0, 1). Y indicates the likelihood that the target user clicks on unseen courses as inferred by the RippleNet model, while P reflects the same likelihood as estimated through the collaborative filtering method.

By integrating the knowledge graph and collaborative filtering course recommendation algorithms from both direct and indirect perspectives, the limitations of using a single approach can be effectively mitigated. The knowledge graph also provides strong interpretability throughout the entire process. The corresponding flowchart of the integrated RippleNet and collaborative filtering recommendation algorithm (RippleNet-CF) is shown in Figure 2.

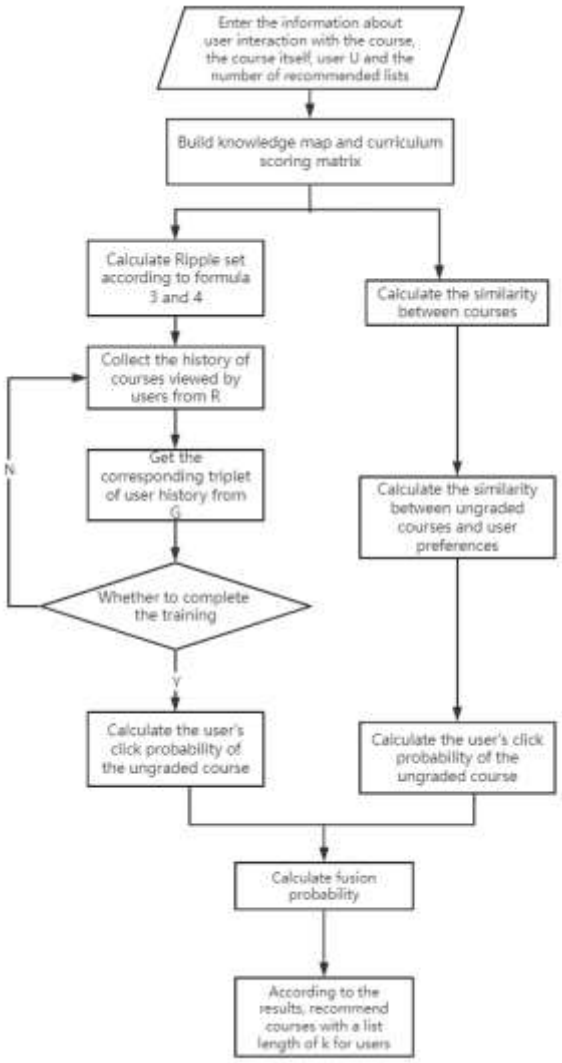


Figure 2. Flowchart of the Integrated Recommendation Algorithm

IV. Experimental Results and Analysis

A. Dataset and Preprocessing

The dataset used in this experiment is MOOCCube, which was collected by a research team from Tsinghua University from the XuetangX platform. They extracted entities such as courses, concepts, and students, and built a knowledge base based on the complex relationships among these entities. This educational resource database is large in scale and rich in data, especially with detailed records of student behavior, including learning duration, frequency, and video segments viewed. The dataset used in this experiment involves nearly 200,000 students and approximately 5 million

video viewing records [14]. Before conducting the experiment, the collected online student dataset needs to be preprocessed. The specific steps are as follows:

- Integrate the video viewing information of each student from the MOOCCube dataset, calculating the total duration of videos for the same course as well as the specific viewing details of the students.
- Handling of missing or duplicate values. For data that is missing or duplicated, it is directly removed.
- The learner's rating is determined by the ratio between their actual viewing time (t) and the total video length (T). That is, the rating score = t/T. Furthermore, these scores are categorized into five distinct levels, with the detailed classification criteria provided in Table 1.

TABLE I. COURSE RATING

Rating	***Score***
S<0.2	1
0.2≤S<0.4	2
0.4≤S<0.6	3
0.6≤S<0.8	4
S≥0.8	5

B. Constructing a Knowledge Graph

Based on the results of data preprocessing, mark the user-course interaction Yuv = 1 if the user's rating for the course is greater than or equal to 4, and mark Yuv = 0 for other scores. According to the courses that users have interacted with, extract the relationships between the attributes of the courses themselves to construct triples. Since there are too many entities in each course for constructing triples, to lower the cost of constructing the knowledge representation, each course is associated with only five extracted entities, as illustrated in Table 2.

TABLE II. EXTRACTION OF SOME COURSE ENTITIES

Course Name	***Entity***
Popular Java Framework	*Tsinghua University Press, October 2018, Lectured by Li Lian, Knowledge Points, Computer*
Data Structures	*People's Publishing House, February 2022, Yu Yun, Knowledge Points, Computer*
Database Principles	*Posts and Telecommunications Press, October 2018, Cao Lan, Knowledge Points, Computer*
Advanced Mathematics	*Tsinghua University Press, September 2018, Zhang Yu, Knowledge Points, Mathematics*

After determining the corresponding entities, construct the corresponding ternary relationships, a total of 5 types of entities are constructed as shown in Table 3.

TABLE III. TERNARY ENTITY RELATIONSHIPS

Entity	***Relationship***	***Entity***
Course Name	*Taught by*	*Teacher*
Course Name	*Published by*	*Specific Publisher*
Course Name	*Time*	*Specific Publication Time*
Course Name	*Belongs to*	*Specific Category*
Course Name	*Contains*	*Knowledge Points*

Based on the construction of ternary relationships for association: for instance, if a teacher teaches several courses, one of the courses can be associated with another by the common teacher who teaches them, as specifically shown in Figure 3: In this experiment, a total of 447,517 ternary relationships were constructed.

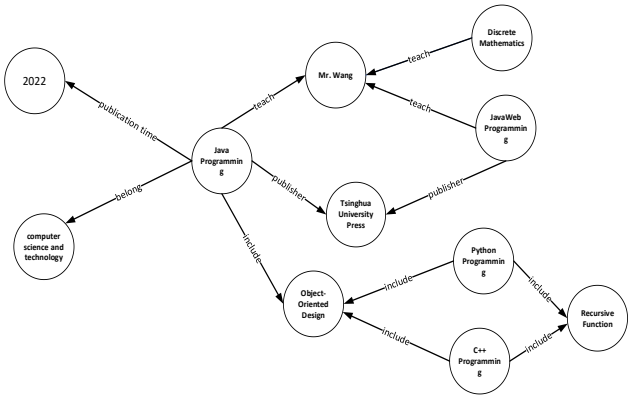


Figure 3. Partial View of the Knowledge Graph

C. Evaluation Metrics

The experimental results in this paper adopt a Top-N recommendation strategy for delivering personalized suggestions to target users. Performance is assessed through three evaluation indicators: precision, recall, and F1 score. In this

context, L(u) denotes the actual recommendation list for user U in the test dataset, while R(u) corresponds to the predicted list generated by the algorithm. Here, U refers to the set of users, and I signifies the collection of available courses.

- Precision: The calculation method is as shown in Formula (8).

$$Precision = \frac{\sum_u \epsilon U |L(u) \cap R(u)|}{\sum_u \epsilon U |L(u)|} \quad (8)$$

- Recall: The calculation method is shown in Equation (9).

$$Recall = \frac{\sum_u \epsilon U |L(u) \cap R(u)|}{\sum_u \epsilon U |R(u)|} \quad (9)$$

- F1 Score (F-Measure): The calculation method is shown in Equation (10).

$$F1 = \frac{2Precision * Recall}{Precision + Recall} \quad (10)$$

D. Experimental Results Analysis

In this paper's RippleNet-CF algorithm, the weight β in equation (9) needs to be trained with corresponding parameters, and the results are shown in Figure 4:

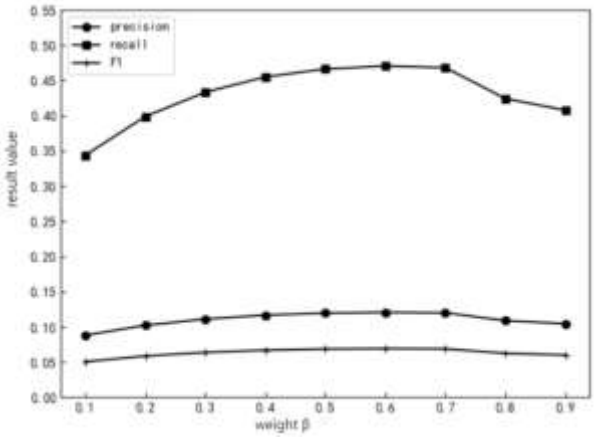


Figure 4. RippleNet-CF Results Chart

From Figure 4, it can be concluded that both accuracy and recall increase as the weight value increases within the range of [0.1, 0.6], corresponding to higher probability values. The accuracy and recall reach their maximum when the weight β equals 0.6. However, the coverage rate is highest at 0.4 and then decreases as the weight value increases.

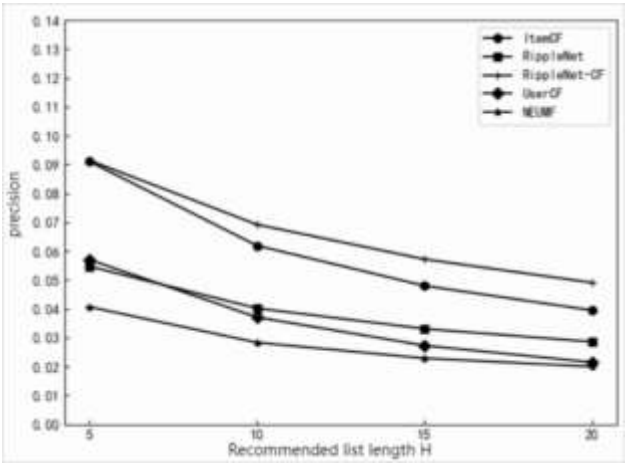


Figure 5. Accuracy Results Chart

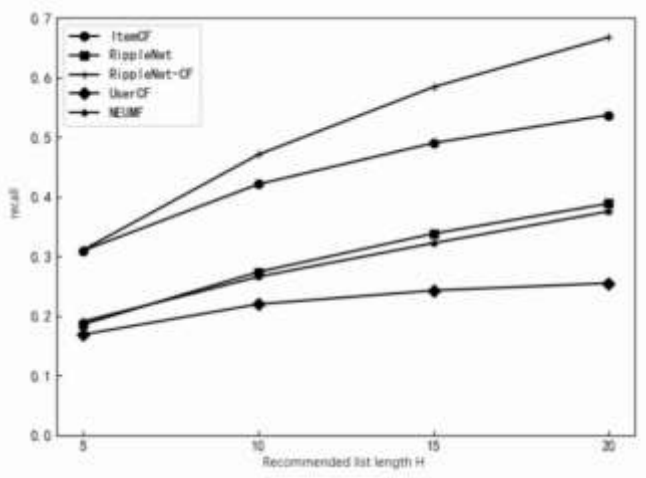


Figure 6. Recall Results Chart

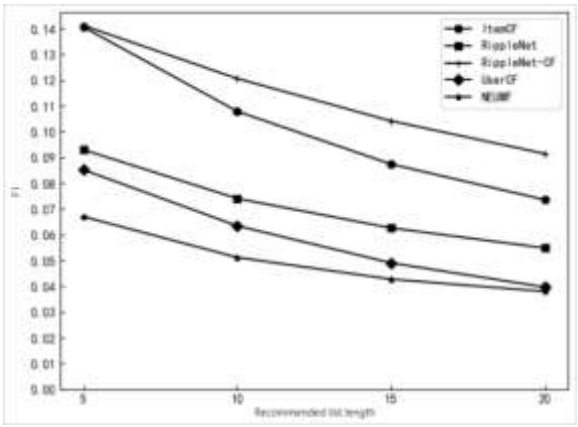


Figure 7. F1 Score Chart

From Figures 5, 6, and 7, it can be seen that at β= 0.6 and h = 2 as the recommendation list increases, the RippleNet-CF method has the best accuracy, recall, and F1 scores compared to the other four algorithms. This is because RippleNet-CF not only uses the interaction information between users and items but also mines the potential connections between courses to expand the information source, thereby improving the optimization effect.

V. Summary and Future Work

In response to the traditional item-based collaborative filtering algorithm, which does not fully utilize the attribute information of items themselves, this paper proposes the RippleNet-CF method using course attribute knowledge graphs and interaction information. This method uses knowledge graphs to explore the potential connections between courses and collaborative filtering to explore existing user connections, thereby improving the issues of data sparsity and cold start problems. However, courses are offered according to semesters and have strong practical sequential characteristics. Future work will consider incorporating time series feature information to further improve accuracy.

VI. Acknowledgment

The authors would like to express their gratitude to Anhui Xinhua University (China) for the support the University-level Scientific Research Project of Anhui Xinhua University (2024zr012). Additionally, we appreciate the support from. Provincial Innovation and Entrepreneurship Program for College Students (S202412216185)

References

[1] S. Wu, F. Sun, W. Zhang, et al., "Graph neural networks in recommender systems: a survey," ACM Computing Surveys, vol. 55, no. 5, May 2022, pp. 1–37, doi:10.1145/3519724.

[2] S. Wang, L. Cao, Y. Wang, et al., "A survey on session-based recommender systems," ACM Computing Surveys (CSUR), vol. 54, no. 7, Aug. 2021, pp. 1–38, doi:10.1145/3460951.

[3] J. Li, Z. Ye, "Course recommendations in online education based on collaborative filtering recommendation algorithm," Complexity, vol. 2020, Apr. 2020, Article ID 8813370, doi:10.1155/2020/8813370.

[4] P. K. Singh, R. Ahmed, I. S. Rajput, et al., "A comparative study on prediction approaches of item-based collaborative filtering in neighborhood-based recommendations," Wireless Personal Communications, vol. 121, no. 6, Nov. 2021, pp. 857–877, doi:10.1007/s11265-021-01696-1.

[5] G. Piao, J. G. Breslin, "A study of the similarities of entity embeddings learned from different aspects of a knowledge base for item recommendations," in Proceedings of the European Semantic Web Conference (ESWC 2018), Springer, Cham, June 2018, pp. 345–359, doi:10.1007/978-3-319-93417-4_21.

[6] M. J. Pazani, D. Billsus, "Content-based recommendation systems," in The Adaptive Web: Methods and Strategies of Web Personalization, Springer, Berlin, Heidelberg, May 2007, pp. 325–341, doi:10.1007/978-3-540-72079-9_10.

[7] H. Wang, F. Zhang, J. Wang, et al., "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," in Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018), ACM Press, Oct. 2018, pp. 417–426, doi:10.1145/3269206.3271764.

[8] W. Jiang, Y. Sun, "Social-RippleNet: Jointly modeling of ripple net and social information for recommendation," Applied Intelligence, vol. 53, no. 3, Mar. 2023, pp. 3472–3487, doi:10.1007/s10489-021-03214-7.

[9] Y. Q. Wang, L. Y. Dong, Y. L. Li, et al., "Multitask feature learning approach for knowledge graph enhanced recommendations with RippleNet," Plos One, vol. 16, no. 5, May 2021, e0251162, doi:10.1371/journal. pone. 0251162.

[10] H. Wang, F. Zhang, X. Xie, et al., "DKN: Deep knowledge-aware network for news recommendation," in Proceedings of the 2018 World Wide Web Conference (WWW 2018), ACM Press, Apr. 2018, pp. 1835–1844, doi:10.1145/3178876.3186143.

[11] H. Wang, F. Zhang, M. Hou, et al., "Shine: Signed heterogeneous information network embedding for sentiment like prediction," in Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM 2018), ACM Press, Feb. 2018, pp. 592–600, doi:10.1145/3159652.3159668.

[12] X. Yu, X. Ren, Y. Sun, et al., "Personalized entity recommendation: A heterogeneous information network approach," in Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM 2014), ACM Press, Feb. 2014, pp. 283–292, doi:10.1145/2556195.2556222.

[13] Y. Cao, X. Wang, X. He, et al., "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in The World Wide Web Conference (WWW 2019), ACM Press, May 2019, pp. 151–161, doi:10.1145/3308558.3313433.

[14] F. M. Harper, J. A. Konstan, "The MovieLens Datasets: History and Context," ACM Transactions on Interactive Intelligent Systems, vol. 5, no. 4, Dec. 2016, Article No. 19, doi:10.1145/2827872.

.