Eng Int Syst (2025) 4: 405–415

Engineering
Intelligent Systems

Deep Learning Model for Image Classification in Machine Vision

Xiaojun Zhang*

School of Artificial Intelligence, Guangzhou Huashang College, Guangzhou 511300, Guangdong, China

In this study, a deep-learning model based on Convolutional Neural Network (CNN) was applied in order to automatically extract image features and improve the accuracy and robustness of classification. A CNN model with multi-layer convolution and pooling structure was constructed, and the ReLU activation function was utilized to improve the ability of nonlinear expression. Multi-category classification was achieved through the Softmax layer. The training dataset was expanded through data augmentation technology to decrease the risk of overfitting and increase the adaptability of the model in complex scenes. A transfer learning strategy was adopted to use the model pre-trained on the large-scale dataset, ImageNet, to decrease the dependence on labeled data, and the model weights were optimized by fine-tuning. Using depthwise separable convolution, a lightweight network structure was implemented to optimize the resource constraints of edge devices. Experimental results showed that the average inference latency of the model was 50.76ms and the average throughput was 20.2 FPS. The model still maintained high classification performance in a low computing environment.

Keywords: Deep Learning; Convolutional Neural Network; Data Augmentation; Transfer Learning; Lightweight Model

1. INTRODUCTION

As artificial intelligence (AI) and deep learning (DL) technology continue to develop rapidly, machine vision has gradually become an important image processing and recognition technology, which is extensively utilized in many fields such as industrial automation, medical diagnosis, intelligent transportation, and security monitoring. The core function of machine vision is to process, analyze, and understand images through computers, so that machines can recognize and make correct decisions. Image classification is a basic task in machine vision, and its accuracy and real-time performance are directly related to the actual effect of the entire application system. At present, traditional image classification methods have significant limitations. Most of the traditional approaches rely on hand-designed features, such as Scale-Invariant Feature Transform (SIFT) [1], Histogram of Oriented Gradients (HOG) [2], etc. These methods work well when processing simple images, but the classification result is often less than optimal when the image has a complex background and uneven lighting, or the target object is occluded or noisy. Manual feature design requires professionals to invest much time and effort, and also has the disadvantages of poor scene adaptability and insufficient robustness. Traditional methods often perform poorly in complex, real-world scenarios. To cope with these problems, the use of DL [3] in image classification has gradually drawn attention. CNN [4] uses a multi-layer neural network structure to automatically extract high-level features from raw images, which not only reduces the dependence on manually designed features, but also greatly improves the accuracy and generalization ability of classification. Therefore, the application of DL models to image classification has become a crucial research direction in the area of machine vision. This paper explores in depth the current status, challenges and improvement plans of this direction.

In the field of image classification, many researchers have proposed different solutions to improve classification results, especially in terms of how to overcome recognition errors in complex environments, how to reduce dependence on data, and how to improve the practical application capabilities of models. Li and Wu [5–6] and other scholars proposed an automatic feature extraction method based on CNN, which

*Address for correspondence: Xiaojun Zhang, College of Computer engineering, Guangzhou Huali College, Guangzhou 511300, Guangdong, China, Email: zhangxiaojun00092@163.com

replaced the traditional manual feature design and extracted more discriminative features in the image through layer convolution operations, making the model more adaptable when facing different image scenarios. Bozinovski and Cai [7–8] and other scholars combined transfer learning to migrate the model trained on a large-scale dataset to the target task, thereby reducing the difficulty of training a small sample dataset. Shorten and Maharana [9–10] and other scholars proposed data augmentation methods, which artificially expanded the scale of the dataset by rotating, flipping, cropping and other transformations on the original image, and further improved the robustness of the model. Although these studies have achieved good results in specific tasks, these methods still show certain limitations when dealing with more complex and varied application scenarios. For example, the transfer learning method proposed by Song and Alzubaidi [11–12] affected the stability of the model when processing images with noise or occlusion. The data augmentation technology of Zheng and Iwana [13–14] also affected the classification accuracy due to the low quality of the generated images. Previous research has improved the performance of image classification models to a certain extent, but there is still room for improvement in terms of the generalizability and data dependence of the model, which also provides an important reference for the new improvement methods proposed in this paper.

To address the limitations of traditional image classification methods and the shortcomings of existing DL models, research in recent years has gradually focused on using more advanced DL technologies and various data processing methods to improve the performance and generalizability of the model. For example, Zhang and Kaushik [15–16] and other scholars significantly reduced the model's dependence on large-scale labeled data by combining data augmentation and self-supervised learning, so that it could also obtain better classification results on smaller datasets. Self-supervised learning applies unlabeled data for training, giving the model more information and improving the recognition ability in classification tasks. Research by scholars such as Zhao and Cao [17–18] added lightweight network structure optimization to CNN, enabling the model to adapt to resource-constrained environments, such as edge devices and mobile devices. This lightweight model significantly reduces computing costs and storage requirements while maintaining high classification accuracy, improving real-time performance. However, these methods also have shortcomings. For example, although data augmentation technology expands data, it applies additional noise. Self-supervised learning reduces the reliance on data annotation while also increasing the difficulty of model training. Hence, this paper proposes a DL model that combines CNNs, data augmentation, and transfer learning to improve the generalization ability, adaptability, and real-time performance of image classification models, and verifies the significant improvement of image classification achieved by the proposed method.

The research objective of this paper is to address the shortcomings of traditional image classification methods in machine vision applications and to propose an improved model based on DL to enhance the accuracy and efficiency of the image classification task. This study uses CNN as the core model and optimizes the image classification process through data augmentation [19] and transfer learning technology [20] to achieve real-time response requirements in complex scenarios. First, data augmentation techniques are utilized to expand the training data, decrease the risk of overfitting, and enhance the model's generalization ability; second, transfer learning can effectively utilize the knowledge of pre-trained models, reduce dependence on a large amount of labeled data, and achieve efficient learning on small sample datasets. Additionally, this paper also reduces the computational complexity of the model through the design of lightweight network structure [21], thereby improving its applicability in resource-constrained environments. This paper organically combines these technologies to construct a more adaptable and stable image classification method. The experiments verify the effectiveness of the method on multiple datasets. The results show that the proposed method is superior to the traditional model in terms of classification accuracy, model robustness, and real-time performance. Finally, the method proposed improves the image classification ability of machine vision, and also provides theoretical and technical support for the application of future intelligent vision systems in complex, real-world scenarios.

2. MODEL DESIGN AND IMPLEMENTATION

2.1 Construction of CNN

In this study, a DL model is built on the basis of CNN to automatically extract image features and improve classification accuracy and robustness. The CNN model is depicted in Figure 1.

In the design of CNN, the core function of the convolutional layer is to extract local features of the input image, such as basic elements including edges and textures. The convolution operation uses multiple convolution kernels to slide on the input image to achieve local perception. Assuming that the input image is I and the convolution kernel is K, the following operation is performed at each spatial position:

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(m, n)K(i - m, j - n) \tag{1}$$

$S(i, j)$ represents the output of the convolution operation at this position, and m and n are the dimensions of the convolution kernel. The dimensions of the convolution kernel are usually set to 3×3 or 5×5 to extract image features at different scales. By stacking multiple layers of convolution kernels, the model can gradually learn the high-level features of the image, from simple edge features to complex pattern structures, forming the feature extraction backbone of the CNN.

The ReLU (Rectified Linear Unit) activation function [22] is applied to the feature map after convolution, which is defined as:

$$f(x) = \max(0, x) \tag{2}$$

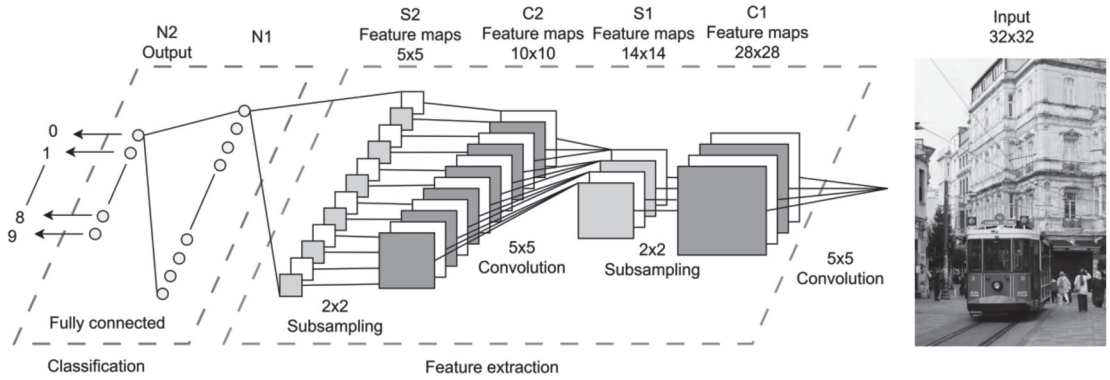

Figure 1 CNN architecture.

The ReLU activation function significantly enhances the model's ability to process complex data by applying nonlinear characteristics by setting negative values to zero. Compared with traditional activation functions such as sigmoid [23] and tanh [24], ReLU has higher computational efficiency, alleviates the gradient vanishing problem, and improves the stability and efficiency of deep network training. Its nonlinear characteristics enhance the network's expressive power and promote the gradual optimization of features across convolutional layers. A maximum pooling layer is often added after the convolutional layer to extract the local maximum through a 2×2 pooling window (with a step size of 2), halving the size of the feature map, which not only retains key information but also reduces computational complexity and enhances the generalization performance of the model. The operation of the maximum pooling layer is:

$$P(i, j) = \max_{m,n} S(i + m, j + n) \tag{3}$$

The pooling layer effectively reduces the amount of computation and memory requirements by reducing the size of the feature map, while increasing the invariance of the features and reducing the sensitivity of the model to small-scale transformations, thereby improving generalization ability and reducing the risk of overfitting.

The output of the fully connected layer is defined as:

$$Y = f(WX + b) \tag{4}$$

The weight W and bias b are continuously updated during the training process, so that the model is gradually optimized to adapt to the data feature distribution. The output of the fully-connected layer is the input of the classification probability, providing the basis for the subsequent classification layer.

The Softmax function is defined as follows:

$$P(y_i) = \frac{e^{Y_i}}{\sum_{j=1}^{C} e^{Y_j}} \tag{5}$$

The output of the Softmax layer ensures that the prediction results of the model are expressed in the form of probability, and the category with the highest probability is the prediction result. This method outperforms the traditional linear discriminant method in classification accuracy, and also outputs the confidence of each category, providing more information for subsequent decision-making.

Batch Normalization [25] is used to standardize the input data of each small batch in order to reduce the internal covariate shift of the model. Batch normalization is calculated with:

$$Z = \gamma \frac{X - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta \tag{6}$$

where μ and σ^2 are the mean and variance of the current batch, respectively, and γ and β are learnable scaling and translation parameters. Through batch normalization, the activation value of the network is kept within a stable range, which helps to speed up the convergence of the model, improve the stability of model training, and further reduce the risk of overfitting. In multi-layer deep networks, batch normalization significantly improves training efficiency.

The CNN constructs an efficient feature extraction and classification framework through the organic combination of multi-layer convolution, ReLU activation, pooling and full connection. The convolution layer and pooling layer extract low-level to high-level features of the image layer by layer; the fully-connected layer achieves the final decision output by integrating features; the Softmax layer provides the probability distribution of multi-category classification. Batch normalization further optimizes the training efficiency in each layer of the network, so that the model can converge quickly and stably. The design of the entire model architecture not only ensures the comprehensiveness and accuracy of feature extraction, but also improves computational efficiency so as to adapt to the needs of complex classification tasks.

2.2 Application of Data Augmentation Technology

To increase the generalization ability of CNN in image classification tasks and its robustness in complex scenarios, a variety of data augmentation technologies are used. These technologies can: expand the scale of the training dataset; reduce the risk of overfitting of the model; and improve the model's adaptability to different environmental conditions.

As depicted in Figure 2, data augmentation is used to generate more samples by transforming the data to enhance the generalization ability of the model. Basic data augmentation

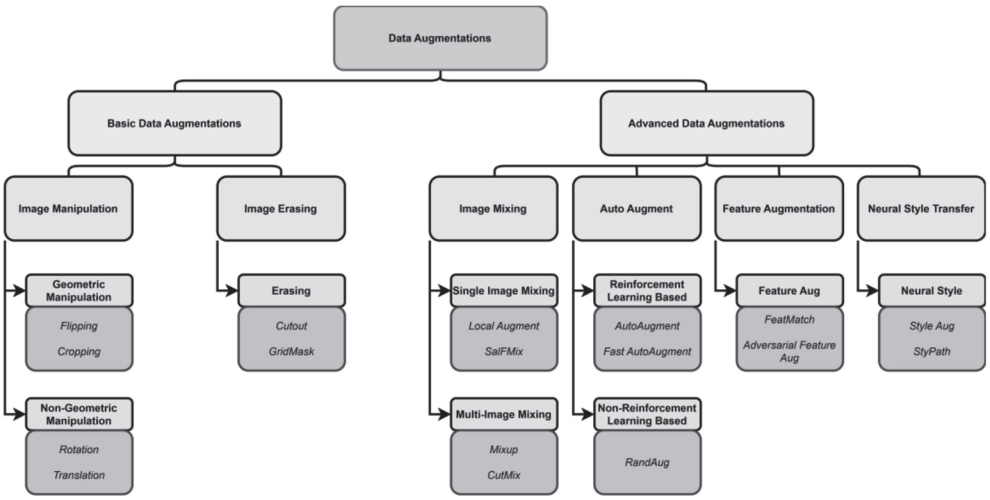

Figure 2 Selection of data augmentation techniques.

methods include geometric transformation (such as rotation, flipping, cropping), color transformation (such as brightness and contrast adjustment) and noise addition. They are simple to use and suitable for most tasks. Advanced data augmentation methods are more complex, such as Cutout [26], which randomly blocks areas in the image, Mixup [27], which mixes two images and labels in proportion, and CutMix [28], which replaces image areas to synthesize samples. GAN (Generative Adversarial Network) [29] generates data to create new samples similar to training data, while Auto Augment [30] and Rand Aug [31] automatically or randomly generate optimal augmentation strategies. Advanced methods are suitable for instances where data is scarce, and can significantly improve model performance.

By implementing image flipping operations, horizontal and vertical flipping can effectively increase the diversity of training samples. For each training image, horizontal flipping is randomly selected to generate new samples, thereby helping the model learn feature expressions in different directions. The flipping operation can be expressed as:

$$I_{\text{flipped}}(x, y) = I(x_{\text{width}} - x, y) \tag{7}$$

where $I(x, y)$ is the original image pixel value; $I_{\text{flipped}}(x, y)$ is the flipped image; x_{width} is the image width. This augmentation method is particularly effective for images with strong object symmetry, so that the model no longer has a bias towards the direction of the image.

Image cropping techniques involve randomly selecting and extracting specific regions from an image, emphasizing key features while retaining image diversity. This approach effectively expands the dataset by generating diverse samples and enhances the robustness of the model. By simulating partial occlusions and changing background environments, this method improves the model's ability to generalize and adapt to changes in real-world visual inputs across different scenarios. In addition, it facilitates the detection and recognition of local features, ensuring more reliable performance in complex environments. The formula for random rotation is:

$$I_{\text{rotated}}(x', y') = I(x, y) \tag{8}$$

where (x', y') is the rotated coordinate. The rotation transformation is implemented through the matrix:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{9}$$

The randomly selected rotation angle is utilized to rotate the image, effectively reducing the model's dependence on a specific perspective, thereby enhancing its adaptability when dealing with unknown scenes. At the same time, using image-cropping technology to randomly crop different areas of the image, attention is focused on the key features of the image while retaining the diversity of samples. This method not only expands the number of samples significantly; it also strengthens the robustness of the model in dealing with partial occlusion and complex background changes, providing the model with a wider range of application possibilities. The formula used for cropping is:

$$I_{\text{cropped}}(x, y) = I(x + \Delta x, y + \Delta y) \tag{10}$$

where Δx and Δy are randomly-selected cropping offsets. The cropped image can be adjusted to a predefined input size to ensure data uniformity and standardization. According to application requirements, the crop size can be dynamically changed within a random range, standardizing the input while simulating a variety of spatial layouts to improve model robustness. The variability of the cropped area improves the model's adaptability to unpredictable application scenarios and maintains stable performance in diverse environments.

Data augmentation techniques such as random scaling and color jittering are applied to generate diverse samples by enlarging or reducing the image to adapt to different scales and scene complexities; color jittering expands data diversity by adjusting brightness, contrast, and saturation, and enhances the model's robustness to lighting changes and color distribution. The scaling factor s is randomly selected in the range of [0.8,1.2]. The formula is:

$$I_{\text{scaled}}(x', y') = I(s \cdot x, s \cdot y) \tag{11}$$

The calculation of color jitter can be expressed as:

$$I_{\text{augmented}} = I \times \alpha + \beta \tag{12}$$

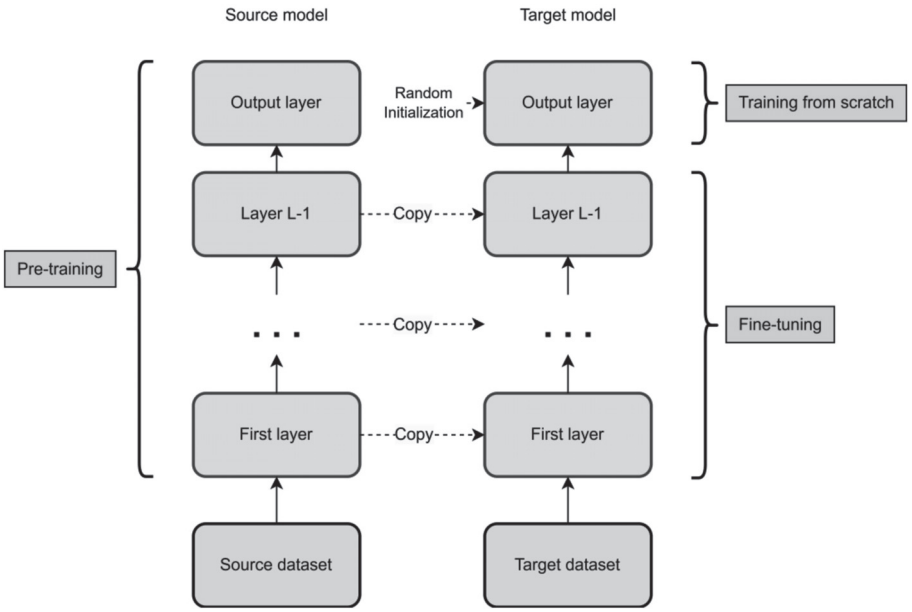

Figure 3 Fine-tuning strategy.

α is the contrast factor, and β is the brightness adjustment value. The selection of adjustment parameters can be achieved through simple random sampling, thereby ensuring the diversity of images during training.

The application of noise addition technology can strengthen the robustness of the model by adding Gaussian noise or salt and pepper noise to the image. This technology enables the model to adapt to classification under low-quality or contaminated data conditions. The expression for adding noise is:

$$I_{\text{noisy}}(x, y) = I(x, y) + \mathcal{N}(0, \sigma^2) \tag{13}$$

The data augmentation technology used constructs a diverse training dataset through a variety of transformation methods. The model obtains a large number of samples, and also improves its robustness and generalization ability in complex scenarios, thereby effectively solving the overfitting problem. During the training process, all these augmentation strategies are randomly applied, so that each training cycle presents different input data, further improving the model's adaptability to unknown data. The successful implementation of this methodology provides a solid basis for the subsequent performance improvement of DL models. By means of these sophisticated augmentation techniques, not only is the models' performance improved; it also demonstrates excellent capabilities in adaptability and robustness.

2.3 Transfer Learning and Pre-Trained Models

ResNet [32] is selected as the base model, and a transfer learning strategy is adopted to fully utilize the deep feature extraction capability of the model obtained by pre-training on the large-scale dataset ImageNet [33] to reduce the need for a large amount of labeled data. The residual learning framework applied by ResNet greatly improves the training performance of deep neural networks. By adding skip connections [34], ResNet allows gradients to be back-propagated more effectively in the network, effectively alleviating the gradient vanishing problem commonly seen in traditional deep network training. This feature makes it possible to build deeper networks, thereby improving feature representation capabilities.

When implementing transfer learning, the pre-trained ResNet model needs to be structurally adjusted to adapt to the specific image classification task of this study. The original fully-connected layer of the model is removed, and, subsequently, a new fully-connected layer suitable for the target classification task is added. Assuming that the target classification task contains C categories, the output dimension of the new fully connected layer is set to C, and its parameters are randomly initialized, as follows:

$$W_{new} \sim \mathcal{N}(0, \sigma^2) \tag{14}$$

where W_{new} is the weight of the newly added fully connected layer, and σ is the selected standard deviation. The parameters of the first few layers of the ResNet model are frozen, and only the newly added fully connected layers are trained. This strategy can retain the low-level features learned by ResNet on ImageNet, while learning the high-level features of specific tasks through the newly added fully connected layers.

During the training process, the loss function used is the cross-entropy loss, which is in the form of:

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}(\theta)) \tag{15}$$

where $L(\theta)$ represents the model loss, and $\hat{y}_{ij}(\theta)$ is the predicted probability of the model. The goal of this loss function is to reduce the classification error rate by updating the weight θ.

As shown in Figure 3, the fine-tuning method [35] is utilized. After freezing most of the layers in the initial training

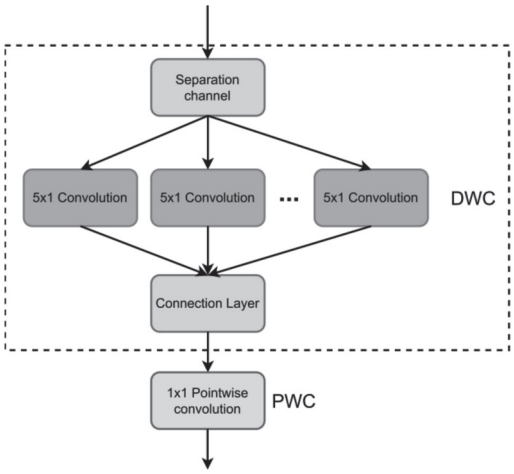

Figure 4 Depthwise separable convolution structure.

phase, training is performed by gradually unfreezing different layers so as to adapt to the features of the new task.

A lower learning rate (1e-4) is used in the fine-tuning phase, ensuring that the pre-trained features of the model are not significantly modified during the training process. The learning rate adjustment of fine-tuning can be achieved through the learning rate decay strategy, as shown in the following formula:

$$\eta_{new} = \eta_{initial} \times \frac{1}{1 + decay \times epoch} \tag{16}$$

η_{new} is the current learning rate; $\eta_{initial}$ is the initial learning rate; decay is the decay factor; *epoch* is the current number of training cycles.

After data augmentation, the number of new samples generated can be expressed as:

$$D_{aug} = D_{orig} + \Delta D \tag{17}$$

D_{aug} is the enhanced dataset; D_{orig} is the original dataset; ΔD is the new sample generated by data augmentation.

The Adam optimizer is utilized to update the parameters, and its update formula is:

$$\theta_{t+1} = \theta_t - \eta \cdot \frac{m_t}{\sqrt{v_t} + \epsilon} \tag{18}$$

m_t and v_t are the first-order moment estimate and the second-order moment estimate, respectively, and ϵ is a smoothing term used to avoid division by zero.

A hybrid loss function is used to combine the cross-entropy loss with the regularization term. The formula is:

$$L_{mixed} = L(\theta) + \lambda R(\theta) \tag{19}$$

L_{mixed} is the hybrid loss; $R(\theta)$ is the regularization term; λ is the hyperparameter for controlling the regularization strength. In this way, the model can not only effectively classify, but also maintain good generalization performance.

In the network training process, model training is regularly monitored, and an appropriate early stopping strategy is established [36] to prevent the performance of the model from degrading on the validation set. By selecting ResNet as the basic model and combining the strategy of transfer learning, the model's feature extraction ability and learning efficiency in specific image classification tasks are optimized, providing a solid foundation for subsequent applications.

2.4 Optimization of Lightweight Network Structure

To improve the operating efficiency of the model in edge devices or real-time applications, the lightweight network architecture EfficientNet is utilized to compress model parameters and computational complexity to reduce resource usage and ensure classification performance in a low computing environment.

EfficientNet [37] adopts a compound scaling strategy to balance the depth, width and resolution of the model. Traditional methods usually expand a single dimension, resulting in inefficient model resource utilization. The compound scaling method adjusts the depth d, width w and input resolution r simultaneously through a unified scaling factor ϕ:

$$d = \alpha^{\phi}, w = \beta^{\phi}, r = \gamma^{\phi} \tag{20}$$

α, β and γ are preset constants. This method ensures the balanced growth of the network in multiple dimensions, thereby effectively improving the performance of the model without wasting computing resources. By means of this compound scaling strategy, EfficientNet performs better than many previous manually-designed networks on the ImageNet dataset, and uses resources more efficiently.

EfficientNet applies Depthwise Separable Convolution [38], which divides the convolution operation into depthwise convolution (DWC) and pointwise convolution (PWC), as illustrated in Figure 4.

In Figure 4, in DWC, each input channel is convolved independently, while PWC uses 1×1 convolution to recombine the output of depthwise convolution. Compared with the computational complexity of traditional convolution:

$$\text{FLOPs}_{\text{conv}} = H \times W \times D_k^2 \times C_{in} \times C_{out} \tag{21}$$

The computational complexity of depthwise separable convolution can be expressed as:

$$\text{FLOPs}_{\text{depthwise}} = H \times W \times D_k^2 \times C_{in} + H \times W \times C_{in} \times C_{out} \quad (22)$$

The application of this structure reduces the computational complexity of the model at the same performance level. To further reduce the storage requirements of the model on edge devices, weight quantization is implemented. During the quantization process, the floating-point weight w is mapped to a fixed-point number $\tilde{w}$, as shown in the following formula:

$$\tilde{w} = \text{round}\left(\frac{w}{scale}\right) \quad (23)$$

where *scale* is the quantization scale, which is usually taken as the maximum value of the weight range. The quantized model can significantly reduce memory usage while maintaining reasonable classification performance. Weight quantization also makes the inference process more efficient and reduces computational complexity.

For model pruning, a channel pruning strategy based on importance score is adopted. By calculating the contribution of each convolution kernel, the convolution kernel with less impact on the output is selected and removed. The importance score can be defined as the absolute sum of the convolution kernel weights:

$$I(w_i) = \sum_{h=1}^{H} \sum_{w=1}^{W} |w_{i,h,w}| \quad (24)$$

When the score of a convolution kernel is lower than the set threshold, pruning is performed to reduce the number of parameters.

The AutoML (Automated Machine Learning) [39] strategy is used to optimize the network structure. By defining the search space S and the optimization target, reinforcement learning or genetic algorithm techniques are used to find the best configuration in the feasible model architecture. The objective function can be expressed as:

$$\theta^* = \underset{\theta \in \mathcal{S}}{\text{argmin}}\, L_{\text{val}}(\theta), s.t. FLOPs(\theta) \leq T \quad (25)$$

In this process, the accuracy of the model is ensured while meeting the constraints of edge computing resources. This strategy greatly improves the applicability of the model in actual application scenarios and ensures the best balance between performance and resource consumption[40–41].

3. MODEL PERFORMANCE EVALUATION AND VERIFICATION

3.1 Cross-Validation

To verify the performance of the constructed CNN model in the image classification task, a scientific experimental environment was set up, and detailed configuration was performed on data preprocessing and training parameters. Before the experiment, in order to ensure the efficient training of the model, NVIDIA Tesla V100 32GB GPU was selected as the acceleration hardware, combined with 128GB memory and Intel Xeon CPU. Ubuntu 20.04 LTS was selected as the operating system; PyTorch 1.9.1 was used as the DL framework; and CUDA 11.1 was installed to support GPU acceleration.

During the experiment, the entire dataset was randomly divided into 5 subsets of equal size to ensure that the category distribution of each subset was consistent so as to reduce the deviation caused by category imbalance. For each round of experiments, one of the subsets was selected as the validation set, and the remaining four subsets were selected as the training set. The training-validation process was repeated.

In each round of training, feature extraction was performed on the training data by means of multi-layer convolution and pooling operations, and the ReLU activation function was used to enhance the nonlinear expression ability. The Softmax layer was used to implement multi-category classification, and the model was optimized on the training set. The accuracy, average IoU (Intersection over Union) score and loss value changes of each round of training were recorded.

After each round of training, the model was tested using the validation set of that round, and the validation accuracy, average IoU, loss value and other indicators of the model were recorded. Through independent validation of each fold in the five rounds of experiments, the final average value was calculated to more accurately evaluate the performance of the model in terms of its generalization.

The results of each round of experiments were recorded in detail, including validation accuracy, average IoU and loss value. The confusion matrix was utilized to analyze the classification effect and observe the recognition accuracy of the model for each category.

The validation accuracy, average IoU and loss value of the model in the 5-fold cross-validation are displayed in Table 1. The average accuracy of the five rounds of validation is 91.12%, and the average IoU score is 0.850, indicating that the model performs consistently in each fold. The average loss value is 0.248, indicating that the model has a good fit on complex datasets.

3.2 ROC Curve

Three different ROC (Receiver Operating Characteristic) curve datasets were constructed by defining the false positive rate (FPR) and the true positive rate (TPR). The datasets FPR1, TPR1, FPR2, TPR2, FPR3, and TPR3 represent the ROC curves of three categories respectively. The trapz function was used to calculate the area under curve (AUC) of each ROC curve.

As illustrated in Figure 5, by analyzing the ROC curves and AUC values of each category, it is concluded that the model has different distinguishing abilities and robustness on the three types of data. The model performs best in category 3, and its curve shape and AUC value both show excellent classification effects; the curve stability and AUC value of category 2 are slightly higher than those of category 1, which is suitable for scenes with flexible requirements; although category 1 is relatively weaker, it still maintains a high AUC

Table 1 Experimental results of cross-validation.

Fold	Validation accuracy (%)	Average IoU	Average loss
Fold 1	91.2	0.851	0.245
Fold 2	90.7	0.846	0.258
Fold 3	91.5	0.854	0.241
Fold 4	90.9	0.849	0.25
Fold 5	91.3	0.852	0.247
Average	91.12	0.85	0.248

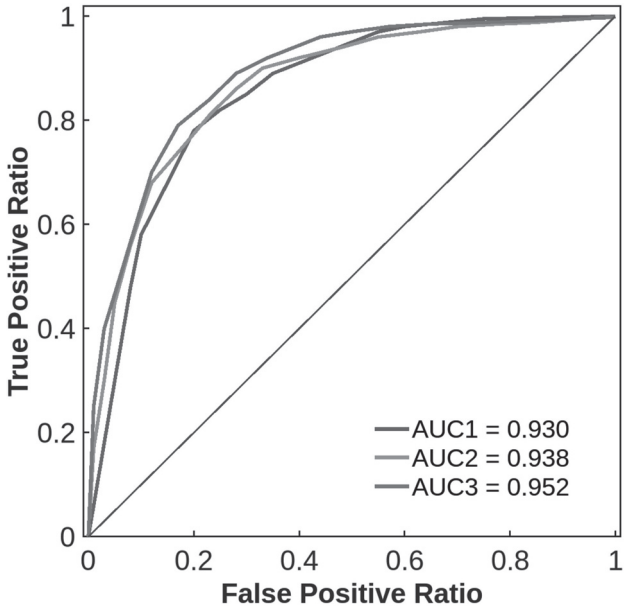

Figure 5 ROC curve.

value, which is suitable for tasks that are tolerant of false alarm rates. The model not only has high overall accuracy and reliability in multi-category classification tasks, but also has certain robustness, and can adapt to complex and changing application scenes.

3.3 Model Stability and Robustness Evaluation

Different intensities of Gaussian noise were added to the data to determine the robustness of the model under different noise interference conditions. The experiment changes the noise intensity to simulate the noise interference of the image under sensor failure, data acquisition error, etc., so as to systematically test the stability and robustness of the model.

As shown in Figure 6, Gaussian noise with different variances of 0.01, 0.02, 0.05, 0.1 and 0.2 is gradually added to the image. As the noise variance increases, the number and intensity of noise points in the image gradually increase, and the quality and details of the image are gradually lost. The impact of noise ranges from subtle to obvious and from imperceptible to significantly visible, and ultimately affects the clarity and accuracy of the image.

Table 2 shows the stability score and classification performance changes of the model under different noise intensities. When the noise intensity is small (such as 0.01), the stability score of the model remains at 0.98. As the noise intensity increases, the stability of the model decreases, but even when the noise intensity is 0.2, the model's stability score can still be maintained at 0.73, which shows its good adaptability under strong disturbances. The model still maintains high stability despite noise interference, fully demonstrating its excellent robustness and anti-interference ability.

3.4 Model Inference Latency and Throughput

By recording the start and end time of each input image processing, the average inference time (unit: milliseconds) is calculated, as shown in Figure 7. The experimental results from the five independent tests show the response time fluctuations of the model under different conditions. The average inference latency of the five tests is 50.76ms, while the inference latency of the model is relatively stable. The minimum and maximum values of the inference latency appear in different tests, with the minimum latency being 47.11ms and the maximum latency being 55.02ms. The fluctuation of the latency value is small, and the model shows good stability under different conditions. This is especially important for application scenarios with high real-time requirements, ensuring that the model can still maintain efficient image classification under high load.

Table 3 presents the throughput test results of the model on edge devices. Each test measures the processing speed of

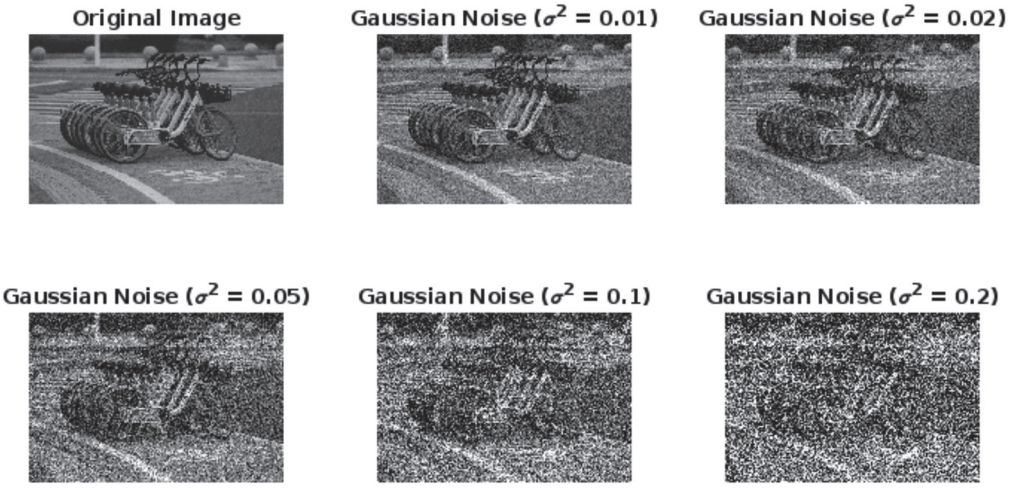

Figure 6 Images with Gaussian noise added.

Table 2 Impact of noise intensity.

Interference type	Noise intensity	Interference impact (%)	Stability score
Gaussian noise	0.01	1.43%	0.98
	0.02	3.24%	0.95
	0.05	5.56%	0.92
	0.1	9.21%	0.84
	0.2	15.48%	0.73

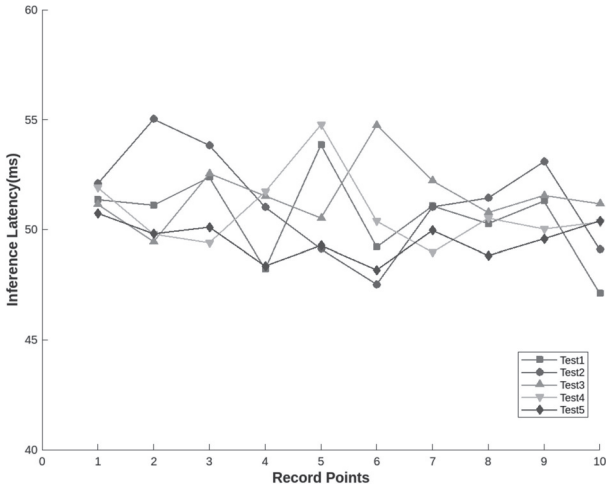

Figure 7 Model inference latency.

Table 3 Model throughput.

Test number	Throughput (FPS)	Min FPS	Max FPS
Test1	20	18	22
Test2	21	19	23
Test3	19	17	21
Test4	20	18	22
Test5	21	19	23
Average value	20.2	18.2	22.2

the model using various image inputs, and provides average, minimum, and maximum throughput values. Across the five tests, the model achieves an average throughput of 20.2 FPS, demonstrating consistent and efficient performance that supports real-time response. For applications that require high throughput, the model can effectively handle high-frequency image inputs. The difference between the minimum and maximum FPS values indicates that the throughput of the model is affected by the complexity of the image input and the hardware load.

4. CONCLUSIONS

This study applied a DL method based on CNN, combining data augmentation, transfer learning and lightweight model optimization to improve the accuracy and adaptability of image classification in the field of machine vision. By constructing CNN with multi-layer convolution and pooling structures, automatic feature extraction was achieved, and the ReLU activation function and Softmax classification layer were used to achieve multi-category image classification. The application of data augmentation technology and transfer learning strategy of pre-trained models can effectively expand the training data and improve the generalization ability and stability of the model in complex environments. At the same time, the lightweight network architecture EfficientNet was adopted to reduce the amount of calculation while ensuring the performance of the model in order to adapt to resource-constrained edge devices.

The findings demonstrate that the proposed method performs well in terms of classification accuracy, real-time and resource utilization efficiency, verifying its application value in complex scenes. However, the robustness of the model to extreme noise and illumination changes still needs to be improved. Future work can further study the application of self-supervised learning and reinforcement learning to enhance the environmental adaptability of the model and optimize the learning ability of the model on large-scale unlabeled datasets, so as to promote the widespread application of intelligent vision systems in diverse environments.

FUNDING

This work was supported by the Office of the Ministry of Education, Provincial First Class Undergraduate Major Construction Site, Computer Science and Technology, Education and Higher Education Department Letter [2022] No. 14; Guangdong Provincial Department of Education, Quality Engineering Project, Yue Jiao Gao Han [2021] No. 29, Information Industry Composite Application Talent Training Plan, 2021; Education Science Planning Project (Higher Education Special) Project No:2021GXJK248; Guangzhou Huali University, Key Discipline, Computer Applications. Guangzhou Huali College, 2021 First Class Undergraduate Course: Database Systems. This project (HS2024SFZY14) is funded under the 2024 Demonstration Specialty Program of Guangzhou Huashang College, focusing on AI talent cultivation; This research is supported by Guangdong Provincial Natural Science Fund Project (2024ZDZX3035), focusing on AI-driven perception in intelligent transportation systems.

REFERENCES

1. Tsourounis D, Kastaniotis D, Theoharatos C, et al. SIFT-CNN: When Convolutional Neural Networks Meet Dense SIFT Descriptors for Image and Sequence Classification[J]. Journal of Imaging, 2022, 8(10): 256–256.
2. Zhou W, Gao S, Zhang L, et al. Histogram of oriented gradients feature extraction from raw bayer pattern images[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2020, 67(5): 946–950.
3. Mohammed A, Kora R. A comprehensive review on ensemble deep learning: Opportunities and challenges[J]. Journal of King Saud University-Computer and Information Sciences, 2023, 35(2): 757–774.
4. Li Z, Liu F, Yang W, et al. A survey of convolutional neural networks: analysis, applications, and prospects[J]. IEEE Transactions on Neural Networks and Learning Systems, 2021, 33(12): 6999–7019.
5. Li X, Li C, Rahaman M M, et al. A comprehensive review of computer-aided whole-slide image analysis: from datasets to feature extraction, segmentation, classification and detection approaches[J]. Artificial Intelligence Review, 2022, 55(6): 4809–4878.
6. Wu D, Zhang C, Wu H, et al. Forest fire recognition based on feature extraction from multi-view images[J]. Traitement du Signal, 2021, 38(3): 775–783.
7. Bozinovski S. Reminder of the First Paper on Transfer Learning in Neural Networks, 1976[J]. Informatica, 2020, 44(3): 291–302.
8. Cai C, Wang S, Xu Y, et al. Transfer learning for drug discovery[J]. Journal of Medicinal Chemistry, 2020, 63(16): 8683–8694.
9. Shorten C, Khoshgoftaar T M, Furht B. Text Data Augmentation for Deep Learning[J]. Journal of Big Data, 2021, 8(1): 1–34.
10. Maharana K, Mondal S, Nemade B. A review: Data pre-processing and data augmentation techniques[J]. Global Transitions Proceedings, 2022, 3(1): 91–99.
11. Song H, Kim M, Park D, et al. Learning from noisy labels with deep neural networks: A survey[J]. IEEE Transactions on Neural Networks and Learning Systems, 2022, 34(11): 8135–8153.
12. Alzubaidi L, Al-Amidie M, Al-Asadi A, et al. Novel Transfer Learning Approach for Medical Imaging with Limited Labeled Data[J]. Cancers, 2021, 13(7): 1590–1590.
13. Zheng Q, Zhao P, Li Y, et al. Spectrum interference-based two-level data augmentation method in deep learning for automatic modulation classification[J]. Neural Computing and Applications, 2021, 33(13): 7723–7745.
14. Iwana B K, Uchida S. An empirical survey of data augmentation for time series classification with neural networks[J]. PLOS ONE, 2021, 16(7): 1–32.
15. Zhang Z, Liu Y, Zhong S. GANSER: A self-supervised data augmentation framework for EEG-based emotion recognition[J]. IEEE Transactions on Affective Computing, 2022, 14(3): 2048–2063.
16. Kaushik V, Jindgar K, Lall B. ADAADepth: Adapting data augmentation and attention for self-supervised monocular depth estimation[J]. IEEE Robotics and Automation Letters, 2021, 6(4): 7791–7798.
17. Zhao H, Wan F, Lei G, et al. LSD-YOLOv5: A Steel Strip Surface Defect Detection Algorithm Based on Lightweight Network and Enhanced Feature Fusion Mode[J]. Sensors (Basel, Switzerland), 2023, 23(14): 6558–6558.
18. Cao Y, Wei T, Zhang B, et al. ML-Net: Multi-channel lightweight network for detecting myocardial infarction[J]. IEEE Journal of Biomedical and Health Informatics, 2021, 25(10): 3721–3731.
19. Fahimi F, Dosen S, Ang K K, et al. Generative adversarial networks-based data augmentation for brain–computer interface[J]. IEEE Transactions on Neural Networks and Learning Systems, 2020, 32(9): 4039–4051.
20. Chen Y, Qin X, Wang J, et al. Fedhealth: A federated transfer learning framework for wearable healthcare[J]. IEEE Intelligent Systems, 2020, 35(4): 83–93.

21. Cao Z, Xu X, Hu B, et al. Rapid detection of blind roads and crosswalks by using a lightweight semantic segmentation network[J]. IEEE Transactions on Intelligent Transportation Systems, 2020, 22(10): 6188–6197.
22. Vershynin R. Memory capacity of neural networks with threshold and rectified linear unit activations[J]. SIAM Journal on Mathematics of Data Science, 2020, 2(4): 1004–1033.
23. Roodschild M, Gotay Sardiñas J, Will A. A new approach for the vanishing gradient problem on sigmoid activation[J]. Progress in Artificial Intelligence, 2020, 9(4): 351–360.
24. Ahmed M S, Zaghrout A A S, Ahmed H M. Travelling wave solutions for the doubly dispersive equation using improved modified extended tanh-function method[J]. Alexandria Engineering Journal, 2022, 61(10): 7987–7994.
25. Garbin C, Zhu X, Marques O. Dropout vs. batch normalization: an empirical study of their impact to deep learning[J]. Multimedia Tools and Applications, 2020, 79(19): 12777–12815.
26. Ding Y, Liu C, Zhu H, et al. A supervised data augmentation strategy based on random combinations of key features[J]. Information Sciences, 2023, 632(1): 678–697.
27. Yang T, Huang Y, Xie Y, et al. MixOOD: Improving Out-of-distribution Detection with Enhanced Data Mixup[J]. ACM Transactions on Multimedia Computing, Communications and Applications, 2023, 19(5): 1–18.
28. F. Jiang, “Animation Design System Based on 3D Image Technology”, Engineering Intelligent Systems, vol. 32 no. 5, pp. 411–419, 2024.
29. Cai Z, Xiong Z, Xu H, et al. Generative adversarial networks: A survey toward private and secure applications[J]. ACM Computing Surveys (CSUR), 2021, 54(6): 1–38.
30. Naghizadeh A, Abavisani M, Metaxas D N. Greedy autoaugment[J]. Pattern Recognition Letters, 2020, 138(1): 624–630.
31. Cubuk E D, Zoph B, Shlens J, et al. Randaugment: Practical data augmentation with no separate search[J]. arXiv preprint arXiv:1909.13719, 2019, 2(4): 7–7.
32. Koonce B, Koonce B. ResNet 50[J]. Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization, 2021: 63–72.
33. Morid M A, Borjali A, Del Fiol G. A scoping review of transfer learning research on medical image analysis using ImageNet[J]. Computers in Biology and Medicine, 2020, 128(4): 104115–104115.
34. Zhou Z, Siddiquee M M R, Tajbakhsh N, et al. Unet++: Redesigning skip connections to exploit multiscale features in image segmentation[J]. IEEE Transactions on Medical Imaging, 2019, 39(6): 1856–1867.
35. Ding Ning, Qin Yujia, Yang Guang, Wei Fuchao, Yang Zonghan, Su Yusheng, et al. Parameter-efficient fine-tuning of large-scale pre-trained language models[J]. Nature Machine Intelligence, 2023, 5(3):220–235.
36. Becker S, Cheridito P, Jentzen A. Deep optimal stopping[J]. Journal of Machine Learning Research, 2019, 20(74): 1–25.
37. S. Liu, " Coupling of Image Authentication and Identification of Logistics Blockchain Based on Cloud Computing", Engineering Intelligent Systems, vol. 33 no. 2, pp. 141–153, 2025.
38. Jang J G, Quan C, Lee H D, et al. Falcon: lightweight and accurate convolution based on depthwise separable convolution[J]. Knowledge and Information Systems, 2023, 65(5): 2225–2249.
39. Karmaker S K, Hassan M M, Smith M J, et al. Automl to date and beyond: Challenges and opportunities[J]. ACM Computing Surveys (CSUR), 2021, 54(8): 1–36.
40. Tian Y, Li X, Zhang H, et al. VistaGPT: Generative parallel transformers for vehicles with intelligent systems for transport automation[J]. IEEE Transactions on Intelligent Vehicles, 2023, 8(9): 4198–4207.
41. Li X, Miao Q, Li L, et al. Sora for scenarios engineering of intelligent vehicles: V&V, C&C, and beyonds[J]. IEEE Transactions on Intelligent Vehicles, 2024, 9(2): 3117–3122.

Xiaojun Zhang was born in Weinan, Shaanxi. P.R. China, in 1980. He obtained a Master’s degree in Engineering from Guangdong University of Technology. He is currently engaged in teaching and research work at Huali College in Guangzhou. He is dedicated to teaching and research in computer control, control algorithms, and computer applications.
E-mail: zhangxiaojun00092@163.com