Research on Traffic Sign Recognition Algorithm Based on Improved Convolutional Neural Networks

Zhe Liu[1,a], Xiao Li[1,b] and Jiaxian Yang[1,c]

[1]*Department of Software Engineering Beijing University of Technology Beijing, China*
a. 806005126@qq.com, b. 382052461@qq.com, c. yjx6133@163.com

Keywords: Traffic sign recognition, convolutional neural network, deep learning, Image preprocessing, overfitting.

Abstract:To improve the accuracy and real-time performance of traffic sign image recognition and reduce the computational cost, a traffic sign recognition algorithm based on an improved convolutional neural network was constructed. Firstly, the data set is pre-processed using image graying processing, limited contrast histogram equalization method and random erasing method to obtain high-quality data set. Then a traffic sign recognition model TSR_ConvNet is constructed. To reduce the occurrence of overfitting, the dropout layer is added and the Label-smoothing regularization method is used. Added the Batch Normalization layer to normalize input batch samples. And by analyzing the effect of the size of the convolution layer filter on the extracted feature map, the convolution kernel of the optimal size is selected. The German traffic sign data set (GTSRB) was used to perform traffic sign classification and recognition experiments. Experimental results show that a recognition accuracy rate of more than 98.74% and a recognition speed of 17 ms per image are obtained on the GTSRB benchmark data set. The traffic sign recognition model TSR_ConvNet constructed in this paper has the characteristics of short training time, good generalization ability, high recognition accuracy, and fast recognition speed.

1. Introduction

Traffic signs carry a large amount of effective road information, which plays an essential role in regulating traffic flow, alleviating traffic congestion, and predicting road conditions to prevent traffic accidents. Traffic Sign Recognition is an vital part of the Intelligent Transportation System[1]. However, the real natural environment is complex and changeable. The accuracy and real-time nature of traffic sign recognition are extremely vulnerable to factors such as the fading and deformation of traffic signs, the complex lighting environment and weather changes, the obstruction of traffic signs by obstacles, and the blurring of pictures caused by car movement.

Traditional traffic sign recognition algorithms are usually based on a combination of artificial feature extraction and computer learning. Traditional feature extraction mainly includes Histogram of Oriented Gradient[2], Scale-Invariant Feature Transform[3], Local Binary Pattern and so on. Traditional classifiers include Support Vector Machine[4], Random Forest Classifier[5], AdaBoost Algorithm[6], and so on. However, this method needs to design different features for different types

of traffic signs. The quality of the artificially designed features often results in large fluctuations in the recognition accuracy and recognition efficiency of traffic sign images, which presents great challenges in practical applications.

In recent years, Convolutional Neural Networks (CNN)[7] has become the main technology for breakthroughs in the field of computer perception. CNN can learn features from a large number of samples without preprocessing, which not only avoids the difficulty of designing handmade features but also trains more features. The current mainstream convolutional neural network frameworks include LeNet-5[8] for handwritten digit recognition, AlexNet[9], VGGNet[10], GoogLeNet[11], ResNet[12], etc. for image classification. Although they can achieve high recognition accuracy, they take a lot of time to train. This paper proposes a traffic sign recognition model based on an improved convolutional eural network. Image preprocessing, network design, and Label-smoothing regularization[13] are used to solve the problem of overfitting[14] and improve the generalization ability of the model.The structure of the model is simplified while ensuring a high accuracy rate so that the model training time is greatly reduced.

2. Convolutional Neural Network

CNN is an algorithm in deep learning. It can effectively reduce the dimensionality of pictures with a large amount of data into small amounts of data and effectively retain the features of the picture, so it is widely used in large-scale image classification tasks. A typical convolutional neural network consists of an input level, convolutional layer, pooling layer, and fully connected layer.

The convolution layer is the core component of the convolutional neural network. Its role is to abstract the implicit correlation in the input traffic sign image through the convolution kernel matrix to perform feature extraction. The original two-dimensional convolution operation is described by the following (1):

$$\begin{aligned} Y(m,n) &= X(m,n) * H(m,n) \\ &= \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} X(m-i,n-j) H(i,j) \end{aligned} \tag{1}$$

where H (m, n) represents the convolution kernel; X (m, n) represents the feature map input to the convolution layer.

Multiple channels and multiple convolution kernels are respectively subjected to two-dimensional convolution to obtain multi-channel outputs, which need to be combined into one channel. The convolution kernel does not flip during the convolution calculation but performs a sliding window related calculation with the input image. Rephrase it as follows:

$$\begin{aligned} Y^{l} m,n &= X^{k}(m,n) H^{kl}(m,n) \\ &= \sum_{k=0}^{K-1} \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} X^{k}(m+i,n+j) H^{kl}(i,j) \end{aligned} \tag{2}$$

Suppose the size of the convolution kernel is I × J, and the size of the feature map of each output channel is M × N, then the calculations of the convolution layer when each sample of this layer performs forward propagation is I × J × M × N × K × L. The learning parameters of the convolution

kernel are I × J × K × L. The ratio of the calculation parameters of the convolution layer to the kernel parameter is CPR = Calculations / Params = M × N. It can be seen that the higher the resolution of the output feature map of the convolutional layer, the higher the CPR, that is, the higher the parameter utilization rate. Convolutional neural networks greatly reduce the number of network parameters by making full use of the idea of local correlation and weight sharing, thereby improving training efficiency.

The activation layer is responsible for activating the features extracted by the convolution layer. Since the convolution operation is a corresponding linear transformation of the input image and the convolution kernel, the activation layer needs to be introduced to perform non-linear mapping on it. In order to accelerate the convergence of CNN, this paper chooses the ReLU function, equation(3) is as follows:

$$f(x) = \begin{cases} 0 \ if \ x<0 \\ x \ if \ x>=0 \end{cases} \tag{3}$$

Pooling is a kind of subsampling, which can reduce the feature maps dimension, thereby reducing the amount of operations. At the same time, the network can better cope with changes such as translation, tilt or distortion of pictures. Currently, the main pooling operations are Max pooling, Average pooling, Stochastic pooling. This paper uses the maximum pooling method, which can retain more image texture information.Equation(4) is as follows:

$$y_j^l = \max\left(y_{j1}^{l-1}, y_{j2}^{l-1}, \cdots, y_{jm}^{l-1}\right) \tag{4}$$

where $y_{j1}^{l-1}, y_{j2}^{l-1}, \cdots, y_{jm}^{l-1}$ represents k × k neighboring neurons connected to the neuron j in the pre-convolution layer and the pooling layer.

After performing convolution and pooling operations on the traffic sign image, the extracted features can be input into the fully linked layer for prediction, and the probability corresponding to each category is obtained. According to the number of types of traffic signs in the GTSRB data set, the output layer consists of 43 neurons.

The loss function in this paper is a negative log-likelihood function. The cost payment in this paper is defined as the average of the loss function of the current batch. The process of convolutional neural network leaning is to minimize the cost function through the gradient descent method. The negative log-likelihood function is described by the following (5):

$$L(r, y) = -[y \ln r + (1-y)\ln(1-r)] \tag{5}$$

where r is the probability obtained for each of the 43 categories through a convolutional neural network; y is the true probability of each of the 43 categories.

The cost function is described by the following (6)(7):

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} l(r^{(i)}, y^{(i)}) \tag{6}$$

$$j(w,b) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} \ln r^{(i)} + \left(1 + y^{(i)}\right) \ln\left(1 - r^{(i)}\right)\right] \tag{7}$$

where w, b are the weights and biases of the output fully connected layer; m is the number of iterations.

3. Improved CNN Traffic Sign Recognition Algorithm

3.1. Network Structure

Based on the LeNet-5 network in the experiment, by optimizing the CNN structure and adjusting the network parameters, a TSR_ConvNet structure with a depth of 12 was constructed. This model can not only better extract the low-level features of the image and effectively transfer the gradient, but also avoid the problem that the CNN algorithm builds a complex model that requires a lot of training time and computing resources. TSR_ConvNet network structure is shown in “Figure 1”.

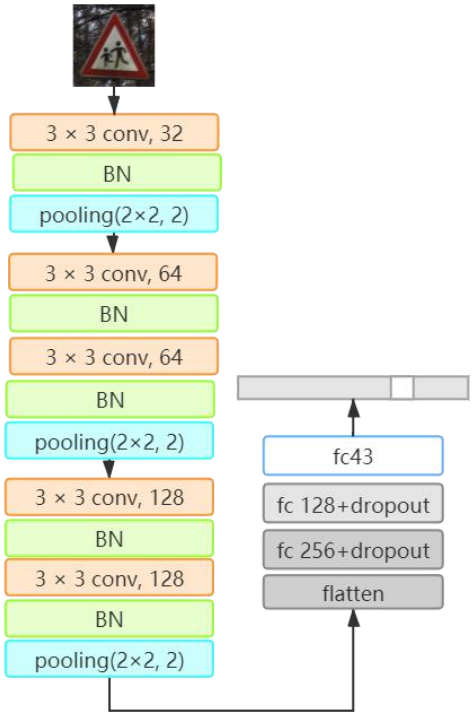

Figure 1: TSR_ConvNet network structure.

The detailed parameters of the TSR_ConvNet network structure are shown in Table 1.

Table 1: TSR_ConvNet network parameters.

Layer	Feature maps & Size	Kernel Size	Stride
Input	32×32×1	-	-
Conv1	32×32×	3×3	1
MaxPooling2	16×16×32	2×2	2
Conv3	16×16×64	3×3	1
Conv4	16×16×64	3×3	1
MaxPooling5	8×8×64	2×2	2
Conv6	8×8×128	3×3	1
Conv7	8×8×128	3×3	1
MaxPooling8	4×4×128	2×2	2
Dense9	256	-	-
Dense10	128	-	-
Softmax	43	-	-

A 3x3 convolution kernel is used in the convolutional layer, and the pooling layer is uniformly 2 × 2. The model adds a Dropout layer and a BN layer. At the same time, the number of output features of each layer is optimized and changed. Through the above optimization, the amount of network training calculations can be effectively reduced, and the CNN training speed and the final recognition rate can be improved.

3.2. Solve the Problem of Overfitting

For convolutional neural networks, when the number of training samples is too small, the number of training times is too large, or the model capacity is too large, the model is prone to overfitting. In order to improve the generalization ability of the model, the proposed solutions are as follows: 1. Preprocess the training set. 2. Add a Dropout layer in the improved model and set an appropriate Dropout ratio. 3. Add the BatchNormalization layer[15] 4. Label-smoothing 5. In the case of many training rounds, add a callback function to make the algorithm stop in time when the training meets certain constraints. The constraints are mainly for loss. When val_loss When stable, the algorithm calls the callback function to stop training in advance.

When training a convolutional neural network, the input distribution of each layer will change, which will make network training difficult, so a smaller learning rate must be used to solve the problem. The BN layer will internally normalize the batch quantity data of each input of the network to normalize the output to the normal distribution of N (0,1). Using the BN layer can improve the learning rate. When the same effect is achieved, the number of iterations is greatly reduced and the convergence speed is improved. In addition, adding a BN layer can play a certain regularization role to prevent problems such as overfitting and gradient disappearance. The specific steps of the normalization algorithm are as follows:

Input values of x over a mini-batch: $\beta=\{x_i \cdots m\}$ .Output: $\{y_i=BN_{\gamma,\beta}(x_i)\}$

Calculate batch means:

$$\mu_\beta \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad (8)$$

Calculate batch variance:

$$\sigma_{\beta}^{2} \leftarrow \frac{1}{m}\sum_{i=1}^{m}\left(x_{i}-\mu_{\beta}\right)^{2} \tag{9}$$

Normalize:

$$x_{i} \leftarrow \frac{x_{i-}\mu_{\beta}}{\sqrt{\sigma_{\beta}^{2}+\varepsilon}} \tag{10}$$

$$y_{i} \leftarrow \gamma x_{i}+\beta = BN_{\gamma,\beta}\left(x_{i}\right) \tag{11}$$

where γ is a scale factor; β is a translation factor.

In the traffic sign classification problem, the last layer is fully connected , and then corresponds to the one-hot encoding of the label. The combination of this encoding method and the method of adjusting the parameters by reducing the value of the loss function will encourage the model to have very different output scores for different categories, that is, the model overly trusts its judgment. Excessive labeling of the model by the model is believed to lead to overfitting. Label-smoothing regularization (LSR) is one of the effective methods to deal with this problem. Its specific idea is to reduce the trust in labels. For example, we can reduce the target value of loss from 1 to 0.9, or rise from 0 to 0.1. It transforms the true probability into:

$$q_{i} = \begin{cases} 1-\varepsilon & if\ i=y \\ \dfrac{\varepsilon}{K-1} & otherwise \end{cases} \tag{12}$$

where ε is a small constant; K is the number of categories; y is the real label of the picture; i represents the i-th category, and q_i is the probability that the picture is the i-th category.

LSR is a regularization method that implements constraints on the model and reduces the degree of model overfitting by adding noise to the label y.

4. Experiment and Result Analysis

4.1. Introduction to the Experimental Platform

The simulation experiment platform used in this article is the Windows 10 operating system, CUDA10.0 and cudnn. The hardware environment is a 6-core Intel (R) Core (TM) i5-9400 processor, clocked at 3.40 GHz, memory is 8G, Nvidia GeForce 1060Ti graphics card, and video memory is 4G. The software uses the integrated development environment Anaconda, Python3 as

the programming language, the Opencv library, and the deep learning frameworks tensorflow2.0 and Kares to train and test the proposed convolutional network model.

4.2. Data Sets Preparation and Preprocessing

In this paper, the German traffic sign data set (GTSRB) collected in the real scene is used as the training data set for the model. The German traffic sign data set contains 43 categories of traffic signs in 5 major categories. The data set is divided into two parts: a training set and a test set, including 39,209 training sets and 12,630 test sets. In this paper, the pictures in the training set are randomly divided into 8: 2 and divided into a training set and a validation data set. The test set is still used as the test set. Some of these traffic signs are shown in “Figure 2”.

Figure 2: Pictures of some traffic signs in the GTSRB dataset.

Although the accuracy rate of the current deep learning-based image recognition algorithms is very high, they are all accomplished with a large number of samples. When there are few training samples or sample types are not evenly clustered. The effect of the algorithm based on deep learning will decline sharply. In order to solve this difficult (the number of pictures of each kind in the dataset is uneven), this paper preprocesses the GTSRB dataset. The overall resolution of the images in the data set is improved, making the images clearer and easier to distinguish. The image pre-processing flowchart is shown in “Figure 3”.

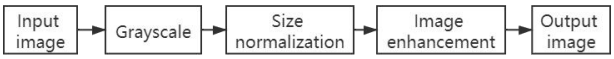

Figure 3: Image preprocessing flowchart.

Image graying: In the process of traffic sign recognition, color is not the main feature such as pictures. Converting color pictures of grayscale pictures can reduce the interference of data and other influences on data. After praying the image, it can effectively reduce the amount of estimation and iterate faster, enhance the training speed of CNN and the recognition effect. In order not to reduce the details of the image, this article uses a weighted average method to grayscale the image. Equation (2) is the grayscale formula used, where R, G, and B are red and green in the color image, respectively. 3 and blue color components, Gray represents the calculated gray value.

$$Gray = 0.3R + 0.59G + 0.11B \tag{13}$$

Image size normalization: The size distribution of the traffic sign images in the GTSRB original data set is uneven, ranging from 15x15 to 250 × 250. According to statistics, the median length and length distribution of the images are 41x40. Considering the amount of computation and image details, the size of the traffic sign image is normalized to 32 * 32.

Image enhancement: ImageDataGenerator method is used to construct an image generator in Keras for data enhancement. By randomly rotating, zooming, moving, cropping and flipping the image, create some new data to balance the distribution of various types of traffic sign images. (2) The Random erasing method is adopted to solve the problem of object obstruction encountered during traffic sign recognition. (3) The histogram equalization algorithm is used to increase the overall contrast of the image, making the image clearer and easier to identify. Some examples of image enhancement are shown in "Figure 4".

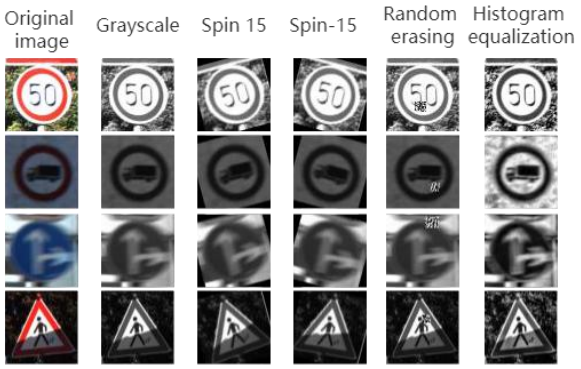

Figure 4: Image enhancement example.

After preprocessing the original sample data according to the above method, the sample quality and sample number are improved, and a high-quality data set is obtained. Image preprocessing effectively improves the recognition accuracy of CNN.

4.3. Experiment Process

The algorithm flow in this paper is mainly composed of three parts. The image is pre-processed first, then CNN is used for feature extraction, and finally the extracted features are sent to the fully connected layer, which is classified by SoftMax classifier. The flow chart of the traffic sign recognition algorithm is shown in "Figure 5":

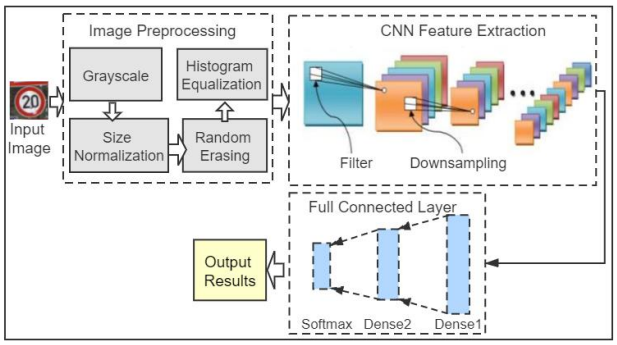

Figure 5: Traffic sign recognition algorithm flow.

4.4. Experimental Results and Analysis

4.4.1. The Comparative Experiment of Multiple Pretreatment Methods

Table 2: Accuracy of various preprocessing methods.

Image preprocessing method	Testing Accuracy /%
No preprocess	89.31
Grayscale	91.87
Data augmentation	93.02
Random erase	92.68
Histogram equalization	95.43

From the experimental results in Table 2, it can be found that when we use the pre-processed data set, the model can achieve a recognition accuracy of 95.43%, which is 6.12% higher than the original image. Proper preprocessing of the data set is crucial to the recognition rate of the network.

4.4.2. Different Dropout Ratio Comparison Experiments

During the network training process, overfitting occurs, that is, the model performs very well on the training set, but it performs incorrectly on the test set. In other words, the training data is overfitted during the training process, which leads to a significant increase in the validation set error rate. Dropout is a regularization method. This method prevents the model from overfitting. The Dropout parameter indicates the ratio of neurons that have been activated in one layer after being passed through the Dropout layer.

Table 3: Comparison of different Dropout ratio test.

Dropout Ratio		Testing Accuracy /%	Training time /h
Convolutional layer	Fully connected layer		
0	0	97.16	1.63
0	0.3	97.58	1.61
0	0.5	98.19	1.58
0.2	0.5	98.74	1.50
0.5	0.5	98.32	1.41

A comparison experiment of different Dropout parameters is shown in TABLE III.As the Dropout parameter increases, the number of discarded neurons increases, and the training time is significantly reduced. Taking the training time and accuracy into consideration, we use a 0.2 in the convolutional layer Dropout ratio and fully connected layer Dropout ratio of 0.5.

4.4.3. Comparison Experiment With or Without Batch Normalization Layer

Table 4: BN layer comparison experiment.

Image preprocessing method	Testing Accuracy /%
No BN layer	98.23
With BN layer	98.74

As can be seen from Table 4, after adding the BN layer, the accuracy of the model has increased from 98.23% to 98.74%, which improves the accuracy of the model.

4.4.4. Convolution Kernel Size Comparison Experiment

In a convolution kernel size control experiment of an improved convolutional neural network model, we use the same convolution kernel size for all convolutional layers. We will test four commonly used convolution kernel sizes: 3 × 3, 5 × 5, 7 × 7. Therefore, a total of 3 models have the same network structure and training parameters except for the different convolution kernel sizes. The experimental results are shown in the Table 5.

Table 5: Convolution kernel size comparison test.

Kernel size	Testing Accuracy /%	Classification time / ms
3×3	98.74	17.00
5×5	98.53	18.43
7×7	98.46	20.35

The experimental results show that when testing on the test data set, using a 3 × 3 size convolution kernel has the best effect.

4.4.5. Label-smoothing Comparison Experiment

Table 6: Label-smoothing comparison test.

Image preprocessing method	Training Accuracy /%	Testing Accuracy /%
No LSR	99.36	97.53
With LSR	98.51	98.74

Experiments show that the accuracy of the training set is higher when there is no LSR, but the model performs slightly lower on the test set. After adding the LSR, the accuracy of the model training set and the test set is equal, which shows that the LSR can solve the model overfitting problem to a certain extent and improve the generalization of the model.

4.4.6. Algorithm Performance Verification in this Paper

“Figure 6” is the final training graph of TSR_ConvNet. Overall, As the number of iterations increases, the training error gradually decreases until it converges smoothly.

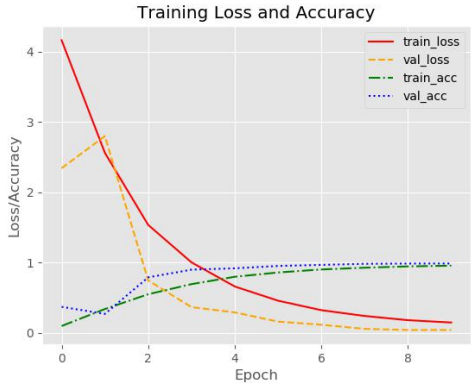

Figure 6: Training Loss and Accuracy

"Figure 6" has 4 curves, of which the red and green curves are the true change curves of the training set loss function and accuracy, respectively, and the yellow and blue curves are the true change curves of the verification set loss function and accuracy, respectively. It can be seen from the figure that the accuracy curve of the training set reaches 98.74% in 10 cycles, and the training is stable. With training, train_loss and val_loss keep falling, and train_acc and val_acc keep rising, reflecting the effectiveness of model training. It takes 1.5h to train the model, and the time to train the model becomes shorter. Using the loaded TSR_ConNet model for traffic sign image recognition, it takes 17ms to recognize an image.

Comparison of accuracy between our algorithm and traditional algorithms in the Table 7.

Table 7: Compared with traditional algorithms.

Image preprocessing method	Testing Accuracy /%
LBP	93.36
HOG+SVM	95.68
Random forests	96.14
Ours	98.74

Comparison of training accuracy and training time between our algorithm and other CNN algorithms in the Table 8.

Table 8: Compared with CNN algorithms.

Algorithm	Testing Accuracy /%	Training time /h
LetNet-5	95.48	0.4
Literature [16]	99.00	11
Literature [17]	98.97	12
Literature [18]	99.65	50
Ours	98.74	1.5

From Table 7, it can be seen that the TSR_ConNet model constructed in this paper has higher recognition accuracy than the traditional algorithm. As can be seen from Table 8,contrast with the existing CNN algorithm, the TSR_ConNet model guarantees a high recognition accuracy while the training time is greatly reduced. Thus, the validity of the TSR_ConNet model is verified.

5. Conclusions

In order to improve the accuracy of traffic sign recognition in natural scenes, a traffic sign recognition algorithm of TSR_ConvNet model is constructed in this paper. A high-quality data set was obtained through image preprocessing. This model can effectively extract features, and has the characteristics of simple structure, small network scale, fast training speed, and strong practicability while ensuring the recognition rate. Multiple control experiments on GTSRB have verified the rationality of the algorithm parameter design in this paper. The experimental results show that the model can attain a recognition accuracy rate of more than 98.74% and a recognition speed of 17ms per image, and the training time of the model is greatly reduced. The next step is to focus on the detection of traffic signs. In a complex natural environment, the traffic signs are accurately detected first, and then the traffic signs are identified to realize the application of traffic sign detection and recognition.

References

[1] Lin Y, Wang P, Ma M. Intelligent transportation system (its): Concept, challenge and opportunity[C]//2017 ieee 3rd international conference on big data security on cloud (bigdatasecurity), ieee international conference on high performance and smart computing (hpsc), and ieee international conference on intelligent data and security (ids). IEEE, 2017: 167-172.

[2] Yao C, Wu F, Chen H, Hao X, Shen Y. Traffic sign recognition using HOG-SVM and grid search[C]//2014 12th International Conference on Signal Processing (ICSP). IEEE, 2014: 962-965.

[3] Lowe D G. Distinctive image features from scale-invariant keypoints[J]. International journal of computer vision, 2004, 60(2): 91-110.

[4] Wang G, Ren G, Wu Z, Zhao Y, Jiang L. A hierarchical method for traffic sign classification with support vector machines[C]//The 2013 International Joint Conference on Neural Networks (IJCNN). IEEE, 2013: 1-6.

[5] Zaklouta F, Stanciulescu B, Hamdoun O. Traffic sign classification using kd trees and random forests[C]//The 2011 international joint conference on neural networks. IEEE, 2011: 2151-2155.

[6] Xu Y, Wang Q, Wei Z, Ma S. Traffic sign recognition based on weighted ELM and AdaBoost[J]. Electronics Letters, 2016, 52(24): 1988-1990.

[7] Cai Z, Cao J, Huang M, Zhang X. Traffic Sign Recognition Based on Convolutional Neural Network[C]//National Conference on Embedded System Technology. Springer, Singapore, 2017: 3-16.

[8] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[9] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[10] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[11] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, et al. Going deeper with convolutions[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 1-9.

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

[13] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the inception architecture for computer vision[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 2818-2826.

[14] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting[J]. The journal of machine learning research, 2014, 15(1): 1929-1958.

[15] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift[J]. arXiv preprint arXiv:1502.03167, 2015.

[16] Zhou S, Liang W, Li J, Kim J. Improved VGG model for road traffic sign recognition[J]. Computers, Materials & Continua, 2018, 57(1): 11-24.

[17] Sermanet P, LeCun Y. Traffic sign recognition with multi-scale convolutional networks[C]//The 2011 International Joint Conference on Neural Networks. IEEE, 2011: 2809-2813.

[18] Jin J, Fu K, Zhang C. Traffic sign recognition with hinge loss trained convolutional neural networks[J]. IEEE Transactions on Intelligent Transportation Systems, 2014, 15(5): 1991-2000.