THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 23, Number 1/2022, pp. 69–78

A FAST TRAFFIC ACCIDENT RECOGNITION METHOD BASED ON EDGE COMPUTING AND DEEP NEURAL NETWORK

Nan WANG[1,2], Qinglu DENG[1], Zeyu JIAO[2], Zhenyu ZHONG[2]

[1] China University of Geosciences, Faculty of Engineering, Wuhan 430074, China
[2] Institute of Intelligent Manufacturing, Guangdong Academy of Sciences, Guangdong Key Laboratory of Modern Control Technology, Guangzhou 510070, China
Corresponding author: Zeyu JIAO, E-mail: zy.jiao@giim.ac.cn

Abstract. Due to the high frequency of occurrence, traffic accidents have always been the main cause of human deaths worldwide, claiming millions of lives every year. Failure to deal with traffic accidents that have occurred in time may cause widespread congestion and even more serious follow-up accidents. Therefore, the development of a system that can quickly recognize traffic accidents will be able to improve transportation efficiency and save more lives. However, limited by complex traffic scenarios and computing resources, most of the existing methods have restricted accuracy in natural environments or are difficult to actually deploy. In this study, a fast traffic accident recognition method based on deep neural network and edge computing is proposed, which only relies on common traffic surveillance cameras. First, images of different traffic conditions are collected to form a traffic condition dataset. Then a deep neural network with an attention module is constructed to automatically recognize traffic accidents in video surveillance. Finally, by parameter pruning and deploying on the edge computing device, the automatic recognition of the four traffic conditions, including dense traffic, sparse traffic, accidents, and burning vehicles, is realized. The recognition accuracy of the proposed method can reach 96.73%, which is of great significance for urban traffic management and human life protection.

Key words: traffic accident, automatic recognition, deep neural network, attention module, edge computing.

1. INTRODUCTION

Due to the high frequency of occurrence, traffic accidents have always been the main cause of human deaths worldwide, claiming millions of lives every year [1]. With the rapid expansion of the scale of cities and the rapid increase in the number of traffic vehicles, traffic congestion and traffic accidents have become the most troublesome problems for traffic management departments [2]. The rapid recognition of traffic accidents can reduce property losses and casualties caused by untimely rescue [3]. The traffic surveillance cameras, which are widely deployed above the road, provide the possibility to identify traffic accidents [4]. Moreover, in the past two decades, traffic surveillance has spread across all corners of urban streets, providing a large number of video data sources for vision-based accident recognition. How to effectively use the existing monitoring data for accident detection is very critical [5]. In view of the use of manpower to monitor each surveillance screen and make judgments may not be enough, because this will consume a lot of human resources.

In the last ten years, with the development of deep learning technology, especially deep neural network (DNN), computer vision technology based on convolutional neural network (CNN) has developed rapidly [6]. DNN does not require manual extraction of image/video features. All features are learned by models through a large amount of data and back propagation (BP) algorithms [7]. Therefore, DNN-based computer vision technology has higher accuracy than traditional image processing methods [8–10]. Bortnikov et al. [11] proposed an accident detection system based on a three-dimensional convolutional neural network (CNN). The generated traffic data is preprocessed by the optical flow method, and noise is injected separately to focus on the motion, and further changes are introduced into the data. By testing real traffic

videos on YouTube, the proposed method can identify traffic accidents. Wang et al. [12] used an improved Faster R-CNN-based regional proposal network to recognize traffic accidents and adopted four strategies to improve the region proposal network. Experimental results show that the mean average precision (mAP) of the proposed algorithm is better than other object detection algorithms. The algorithm has good performance in small object detection. Among them, the object detection speed is 6 frames per second (FPS), which is faster than other object detection algorithms. Le et al. [13] designed an accident detection network called Attention R-CNN, which consists of two streams: one is target detection with classes, and the other is feature attribute calculation. As an attention mechanism for obtaining scene context information, it integrates the global context in the scene into the target detection flow. This introduced attention mechanism enables us to recognize the characteristic attributes of objects. A large number of experiments on the newly constructed data set prove the effectiveness of our proposed network. However, due to the complexity of the natural environment, surveillance-based traffic accident detection is a very necessary and challenging visual task [14].

In addition, computing resources are also one of the core factors restricting the deployment of the accident recognition system [15]. Existing DNN-based models are usually deployed in the cloud or on a server with GPU to satisfy the real-time processing of traffic surveillance data [16]. Song et al. [17] proposed a cloud computing-based intelligent transportation system traffic emergency scheduling model for the traffic emergency dispatching of the traditional intelligent transportation system and used an improved genetic algorithm to solve the model and obtain the optimal dispatching plan. For the monitoring system of the intelligent road environment, Finogeev et al. [18] proposed a fusion model based on cloud computing to process large sensor data and established a comparison between the time series of traffic accidents and the time series of meteorological factors. The cloud computing-based traffic accident recognition method has brought the dawn of solving the problem, but since there are usually tens of thousands of traffic surveillance cameras in a city [19], monitoring so many cameras at the same time brings a great burden to the data transmission network and the cloud recognition model. Although the work of [20] focuses on the use of lightweight web frameworks, cloud transmission still restricts the actual deployment of existing methods. Integrating a certain amount of computing power on the camera can turn the camera into an edge computing device, and then have the ability to automatically recognize the occurrence of a traffic accident. This makes transmitting only the processing results a feasible solution to the above-mentioned challenges.

Edge computing, first proposed by Akamai in 1998 [21], is a distributed computing architecture that moves the computing of applications, data, and services from the central node of the network to the edge node of the network logic for processing [22]. Edge computing decomposes large-scale services that were originally handled by central nodes, cuts them into smaller and easier-to-manage parts, and distributed them to edge nodes for processing. The edge node is closer to the user terminal device, which can speed up the data processing and transmission speed and reduce the delay. Deploying DNN on edge computing devices for real-time processing of traffic surveillance videos on the camera side has become a potential solution for the automatic recognition of traffic accidents. In this study, a fast traffic accident recognition method based on edge computing and DNN is proposed. First, surveillance videos under different traffic conditions are collected to form a dataset. Then, data augmentation is adopted to make the trained model better adapt to complex traffic scenarios. Next, a DNN with an attention module is constructed to automatically recognize traffic accidents in different scenarios. Finally, the trained model is compressed after parameter pruning so that it can be deployed on edge devices. The main contributions of this paper are summarized as:

- A fast traffic accident recognition method based on edge computing and DNN is proposed to recognize traffic accidents in time.
- An attention module is added to DNN to more accurately recognize traffic accidents from video surveillance.
- Parameter pruning is utilized to compress the trained model to reduce the storage space and computing resource requirements of the model, making it easy to deploy on edge computing devices.
- The proposed fast recognition method for traffic accidents combines edge computing devices and DNN, which solves the challenges of insufficient cloud resources and data transmission limitations and gets rid of the limitation of the number of access devices.

2. MATERIALS AND METHODS

This section introduces the adopted hardware devices and the proposed method in detail. First, detailed information about the camera and edge computing device used is given to illustrate the basis of this study. Then, the proposed method is explained point by point. The overall scheme of the proposed method is illustrated in Fig. 1.

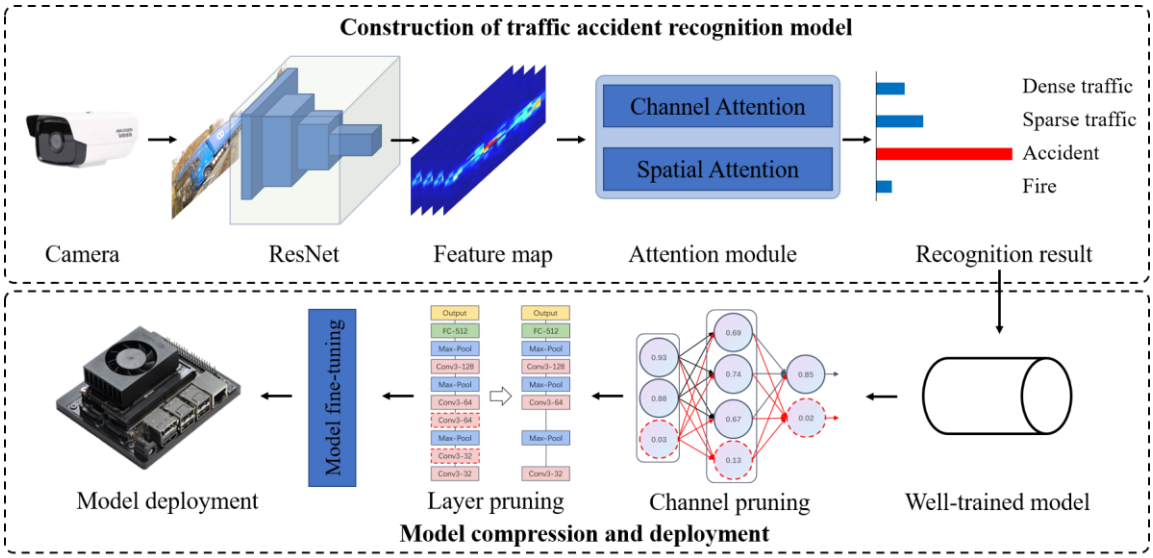

Fig. 1 – Overall scheme of the proposed method.

2.1. Hardware devices

As shown in Fig. 2, this study adopted HIKVISION webcams with 1/2.7" Progressive Scan CMOS sensor as traffic surveillance cameras, which are connected to the remote model server through an integrated RJ-45 Ethernet interface and can capture images with a resolution of 1280×720. The compressed traffic accident recognition model was deployed on a small and cheap edge device, Jetson Xavier NX (NVIDIA Inc, City, CA, USA), with 384 NVIDIA Volta CUDA cores and 48 tensor cores, which only provides 21TOPS of AI computing power.

Fig. 2 – HIKVISION Camera and NVIDIA Jetson Xavier NX.

2.2. Data preprocessing

Some public traffic surveillance datasets have been released recently in the context of the research on traffic accidents surveillance data. In this paper, Traffic-net dataset [23], which contains images of dense traffic, sparse traffic, accidents, and burning vehicles, was utilized as the basis for constructing a traffic accident recognition dataset. There are a total of 4,400 images in different traffic conditions in the original dataset. There are 1,100 images of dense traffic, sparse traffic, accidents, and fire, respectively, 900 of them were used for training and 200 were used for testing. Although these data were collected from the real traffic

environment, most of the collected images were taken under bright light or high definition. However, traffic accidents often occur under different lighting conditions, and the images captured by the camera may also be disturbed by noise. Therefore, data augmentation is exploited to increase the amount of training data and consider various real situations to enhance the robustness of the model. As shown in Fig. 3, 15 different data augmentation methods, including cutout, noise, brightness, etc., were used to expand the dataset. In this way, the original 3,600 training images have been expanded to 57,600.

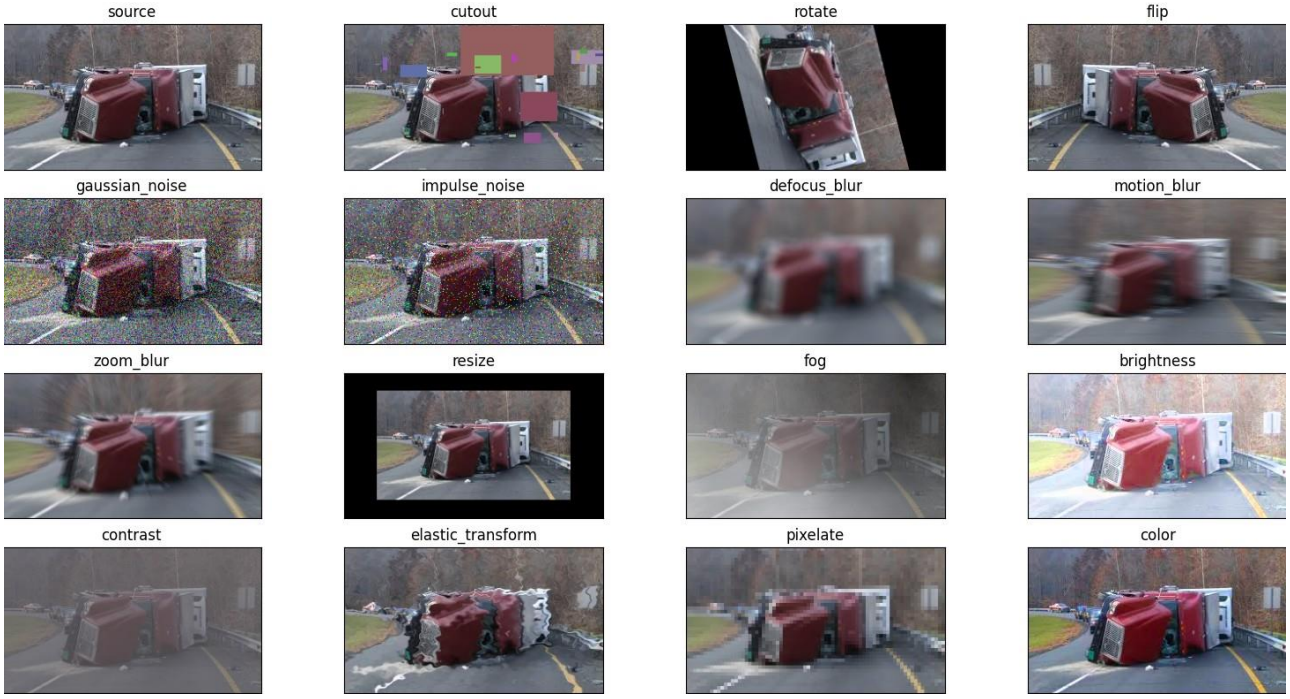

Fig. 3 – Examples of the data augmentation of the original traffic dataset.

2.3. DNN with attention module

After completing the construction and preprocessing of the dataset, a DNN composed of a CNN and an attention module is constructed, as shown in Fig. 4. Among them, CNN is utilized to extract image features of traffic surveillance frames, including texture, contour, color, etc. in the image, and generate high-dimensional vectors that can be processed by classifiers such as multi-layer perceptron (MLP). ResNet [24] was selected as the backbone network because of its advantages of better-fitting classification functions to obtain higher classification accuracy and solve the optimization training problem when the number of layers is deepened.

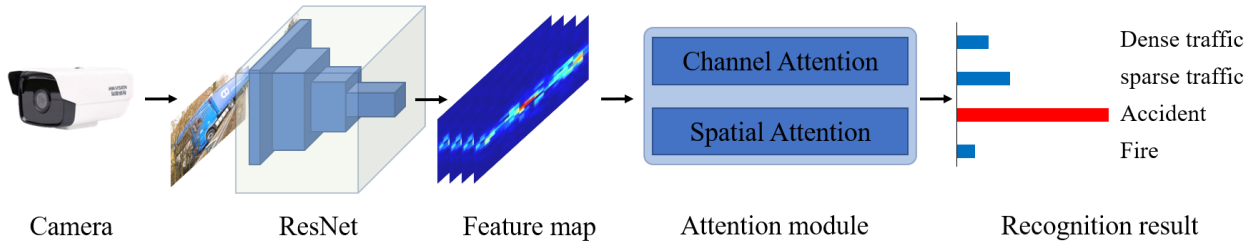

Fig. 4 – Architecture of the ResNet with attention module.

However, due to the limitation of the size of the convolution kernel, although the CNN model can increase the receptive field to a certain extent with the help of operations such as pooling, CNN cannot always pay attention to the global information of the image. Therefore, the attention mechanism was

proposed to adaptively assign importance weights between different regions of the image. This research follows the idea proposed by Woo et al. [25] and built an attention module (as shown in Fig. 5) to allow the model to ignore irrelevant information and pay more attention to the key information in the traffic surveillance frames.

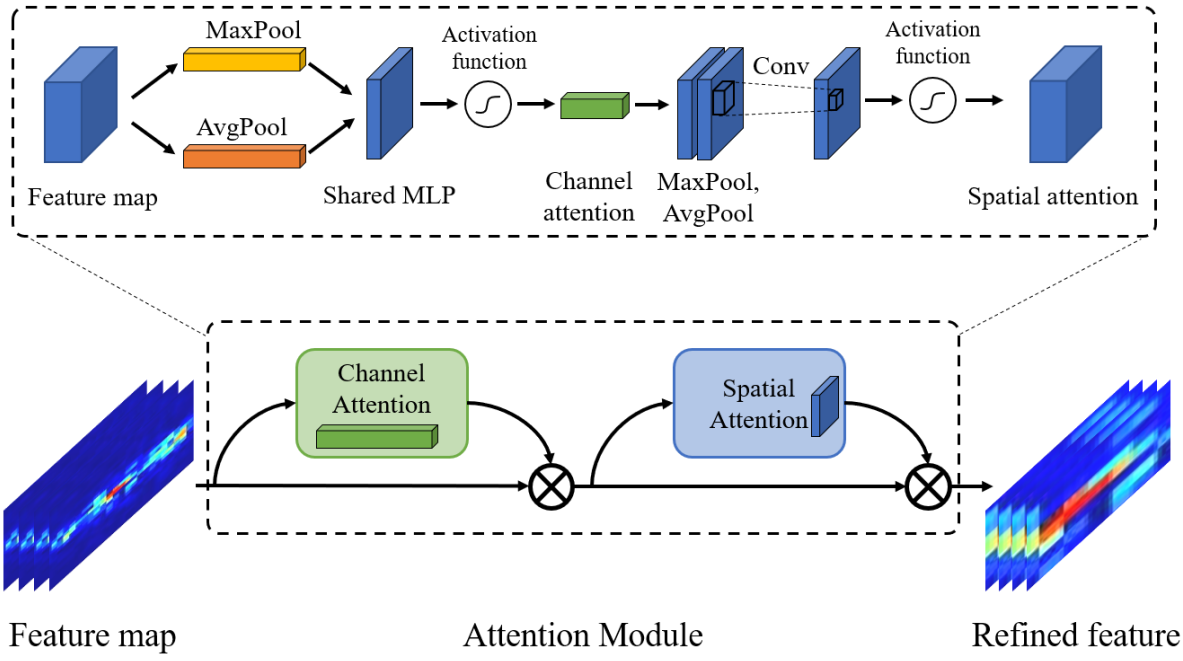

Fig. 5 – Architecture of the attention module.

The attention module is mainly composed of channel attention and spatial attention. These two parts are essentially mapping functions, which can be represented by M_c and M_s respectively. The input of the attention module is the feature maps $\mathbf{F} \in \mathbb{R}^{C \times H \times W}$ extracted by ResNet, and the output $\mathbf{F}'$ of the channel attention can be expressed as

$$\mathbf{F}' = \mathbf{M}_c(\mathbf{F}) \otimes \mathbf{F}, \tag{1}$$

where $\otimes$ denotes element-wise multiplication.

Then, the output $\mathbf{F}'$ of the channel attention is feed into the spatial attention. The final output $\mathbf{F}''$ of the attention module is obtained by mapping, and it is defined as

$$\mathbf{F}'' = \mathbf{M}_s(\mathbf{F}') \otimes \mathbf{F}'. \tag{2}$$

The entire attention mechanism module is composed of basic operations such as multiplication and addition, so the gradient is derivable in the entire model, so it can be trained end-to-end.

2.4. Parameter pruning

The well-trained model leverages a multi-layer network to enable the model to have strong enough feature extraction capabilities to realize the recognition of traffic accidents, but this also makes the model require more computing resources. Therefore, in order to be deployed on edge devices with limited resources, specific operations need to be performed to reduce the model's resource requirements while ensuring that the recognition accuracy rate does not significantly degrade. A naive idea is to reduce the number of parameters in the model by means of parameter pruning. The parameter pruning operation on the well-trained model is mainly composed of two parts: channel sparsity and channel pruning [26].

Since each layer of the model sets a lot of channels to extract different image features, different channels have different effects on the final result. Therefore, the essence of channel sparseness is to polarize the weights of different channels on the final result by setting specific constraints, that is, important

channels get greater weights, and the original unimportant channel weights tend to be close to 0 or equal to 0. If the number of channels is directly reduced, the weights of each channel will change, but it is difficult to achieve the expected weight polarization, and the network feature extraction capability may be weakened due to insufficient channels. Therefore, by introducing a sparse factor γ into the loss function, which acts on the original cross-entropy loss function, the weights of each channel tend to be polarized during each training iteration. The final loss function is calculated as follows:

$$\text{Loss} = \text{cross}_{\text{entropy}}\left(P_i, Y_i\right) + \lambda \sum_{\gamma \in \Gamma} g(\gamma), \tag{3}$$

where $\text{cross}_{\text{entropy}}\left(P_i, Y_i\right)$ is the cross-entropy loss function, which is commonly used in CNN training, P_i indicates the predicted result of the i^{th} input, Y_i denotes the corresponding ground truth label, λ is a weight coefficient used to balance cross-entropy loss and sparsity item, $g(\bullet)$ represents the sparsity item, which is essentially a function with a sparse factor as input, adding on the total loss to guide the training of the network.

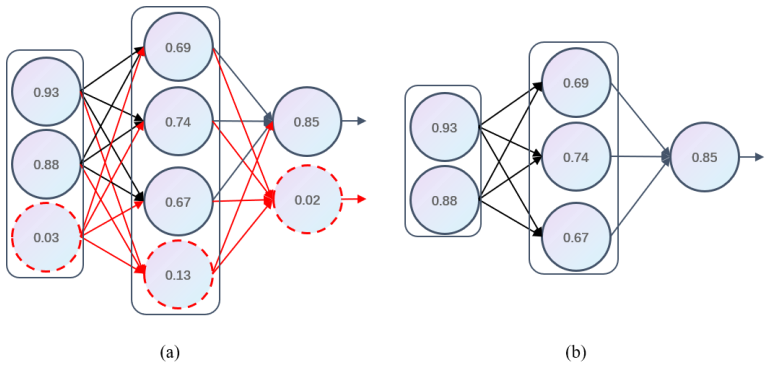

Fig. 6 – Principles of channel pruning.

After the channel sparsity, the weights of different channels on the final result are polarized, and then the channels with less influence on the final result can be pruned to reduce the amount of network parameters. The principle of the channel pruning algorithm is shown in Fig. 6. The coefficient of the batch normalization (BN) layer is utilized to calculate the contribution score of the channel to the network. Then, according to the distribution of pruning rate and coefficients, the channels with high contribution scores are retained, and the channels with low contribution scores are deleted (as red dotted line shown in Fig. 6). When connecting the inter-layer channels, the channels with lower contributions do not participate in the connection, and a simplified model that takes up less storage space is generated. In this process, a pruning threshold compression rate α can be preset as a criterion for judging whether to keep or remove the relevant channel. This threshold is calculated as follows:

$$\theta'^{(k)} = \text{sort}_{\alpha}^{(k)}(\theta)_{+}, \tag{4}$$

where θ and $\theta'^{(k)}$ represent the parameters of the k^{th} BN layer before and after pruning, respectively. $\text{sort}_{\alpha}^{(k)}(\theta)_{+}$ indicates the weight of all channels of the k^{th} BN layer sorted from large to small and removes the smaller part according to hyperparameter α.

The entire optimization process can be represented by pseudocode Algorithm 1.

Algorithm 1 Optimization algorithm of the proposed method

Input: Training image set T, labels Y, learning rate ε and network parameters θ, maximum training iteration I, Hyperparametric compression rate α and sparsity training rate γ, Weight coefficients in the loss function λ

Output: Original network parameters θ, Compressed network parameters θ';

1: **Initialize:** the parameters θ of the ResNet with attention module with pretrained weights θ_1 and $i = 0$

2: **while** *not converged* and $i < I$ **do**
3: $i = i + 1$;
4: **for** $i \leftarrow 0$ **to** I **do**
5: Feed a batch image B_i from the training image set T into the network, the predicted results P_i: $P_i = F_\theta(B_i)$;
6: Calculate the loss of the predicted results according to Eq. (3);
7: Calculate the backpropagation gradient with the aid of the backpropagation algorithm: $\Delta\theta = BP(Loss)$
8: Obtain the optimized network parameters: $\theta = \theta - \Delta\theta \cdot \varepsilon$;
9: **end for**
10: Select network parameters of any layer *k* and sort each weight value from largest to smallest, according to the compression rate α, only the part with the larger value is retained according to Eq. (4);
11: **end while**

2.5. Evaluation metric

The performance evaluation metric is the model size, precision of accident recognition, recall rate, and F1- score. Precision is the final measure of the prediction result, which can be obtained by dividing the true value by the sum of the true and false values:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{5}$$

Recall is a measure of how well each unique label matches the predicted result, which is the true result divided by the sum of the true and false negative values:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{6}$$

F1-score is the harmonic mean of precision and recall, where F1-score achieves the best value at 1 and the worst value at 0. The F1-score is calculated as follows:

$$F1-\ \text{score} = \frac{2(\text{Precision} * \text{Recall})}{(\text{Precision} + \text{Recall})}. \tag{7}$$

3. EXPERIMENT

With the help of the method proposed in Section 2, a DNN with an attention module was first constructed based on the Traffic-net dataset. By adopting the parameter pruning method proposed in the previous section, the constructed model was compressed to facilitate deployment on edge computing devices. After the trained model and the compressed model are evaluated on the testset respectively, the performance of the proposed method is presented numerically. In this section, the experimental details and results are demonstrated separately.

3.1. Experimental detail

The initial learning rate is 10^{-3}, and the decay rate is set to 10^{-1} per 2 000 iterations, the activation function of the hidden layer is the rectified linear unit (ReLu) function, and the activation function of the output layer is the linear function. The maximum iterations of the model were set to 80 epochs. The number of layers and channels of the backbone network follows the settings of [25], and a ResNet network with 50 convolutional layers is used. The batch size is preset to 32 to ensure that the graphic memory can withstand without sacrificing accuracy. In all our experiments, we initialize all channel scaling factors γ to be 0.5, and the weight value λ in the loss function is fixed to 10^{-5}. Compression rate α is set to 0.90, which means that

only about 10% of channels are reserved. Besides, in the experiment, $g(s)=|s|$, which is called L1-norm and is widely used to achieve sparsity. The features extracted by max pooling and average pooling in the spatial attention module are fused by a 7×7 convolutional layer. All the experiments are conducted on an Intel i7-6700 CPU at 4.0 GHz with 16 GB RAM and 8 Nvidia P100 GPU with 16 GB memory. The programming language is Python 3.6, and the integrated development environment is Anaconda 3.

3.2. Experimental result

To further clarify the role of each module in the proposed method, ablation experiments are performed to demonstrate the differences between different methods, as shown in Table 1. It should be noted that the methods compared in Table 1 were all compared under the same epochs and the same hyperparameter settings.

Table 1

Comparison of performance between different methods

Description	Size	Precision	Recall	F1-score
ResNet-50 (baseline)	276.8M	90.30%	92.14%	91.21%
ResNet-50 + Attention	282.9M	97.09%	98.47%	97.78%
ResNet-50 + Attention+pruning	20.4M	96.73%	95.44%	96.08%

The experimental results show that by adding the attention module, the proposed method has a nearly 7% improvement over the baseline, but the model size is only increased by 6M. But after the parameter pruning, the size of the model is reduced by 92.79%, only about 20M in size, but only with a loss of accuracy of 0.36%. This is mainly because the fact that when models such as ResNet were initially constructed, many layers of convolutional layers were stacked to ensure that CNN has strong feature extraction capabilities. However, after training and fine-tuning were completed, there were a large number of channels and layers in the model that will not have a significant impact on the final result, so they can be removed by pruning. After pruning and compression, the model is only about 7.2% of the original model, and the number of parameters and the number of network layers is greatly reduced so that the model can be deployed on edge computing devices with less powerful computing power. By testing the models deployed on edge devices, the recognition accuracy of traffic accidents can also reach 96.73%, and the processing speed can reach 26 FPS. This is sufficient to meet the deployment needs of daily traffic surveillance cameras. As shown in Fig. 7, a visual example is given to better explain the results of the experiment.

Fig. 7 – Visualization of traffic accident recognition results.

4. CONCLUSION

Aiming at the challenges faced by the actual deployment of video surveillance in existing traffic systems, this study proposed a fast traffic accident recognition method based on edge computing and DNN. Through data augmentation, a dataset that is more robust to the natural environment is generated on the basis of the original Traffic-net accident dataset. In addition, by combining CNN and the attention module, the accuracy of traffic accident recognition is effectively improved. Finally, the model is compressed through parameter pruning so that the model can be deployed on edge computing devices. Experimental results show that the proposed method can achieve a recognition accuracy of 96.73% and run-on edge computing devices at a speed of 26 FPS. With the help of the proposed method in this study, traffic accidents can be automatically, quickly, and accurately recognized, which is of great significance for urban traffic management and the protection of human lives.

ACKNOWLEDGEMENTS

This work is supported by the financial support from Key-Area Research and Development Program of Guangdong Province(2018B010108006) and GDAS' Project of Science and Technology Development (No. 2021GDASYL-20210103090).

REFERENCES

1. A. MOHAMMED, K. AMBAK, A. MOSA, D. SYAMSUNUR, *A review of traffic accidents and related practices worldwide*, The Open Transportation Journal, **13**, *1*, pp. 65–83, 2019.
2. D. SINGH, C. MOHAN, *Deep spatio-temporal representation for detection of road accidents using stacked autoencoder*, IEEE Transactions on Intelligent Transportation Systems, **20**, *3*, pp. 879–887, 2018.
3. J. XIA, Y. LIU, D. ZHAO, Y. TIAN, J. LI, Y. ZHONG, N. ROY, *Human factors analysis of china's confined space operation accidents from 2008 to 2018*, Journal of Loss Prevention in the Process Industries, **71**, Art. 104480, 2021.
4. N. KHAN, N. JHANJHI, S. BROHI, R. USMANI, A. NAYYAR, *Smart traffic monitoring system using unmanned aerial vehicles (UAVs)*, Computer Communications, **157**, pp. 434–443, 2020.
5. Z. YANG, L. PUN-CHENG, *Vehicle detection in intelligent transportation systems and its applications under varying environments: A review*, Image and Vision Computing, **69**, pp. 143–154, 2018.
6. Z. JIAO, G. JIA, Y. CAI, *A new approach to oil spill detection that combines deep learning with unmanned aerial vehicles*, Computers & Industrial Engineering, **135**, pp. 1300–1311, 2019.
7. G. HINTON, S. OSINDERO, M. WELLING, Y. TEH, *Unsupervised discovery of nonlinear structure using contrastive backprop- agation*, Cognitive Science, **30**, *4*, pp. 725–731, 2006.
8. X. FENG, Y. JIANG, X. YANG, M. DU, X. LI, *Computer vision algorithms and hardware implementations: A survey*, Integration, **69**, pp. 309–320, 2019.
9. Z. JIAO, H. LEI, H. ZONG, Y. CAI, Z. ZHONG, *Potential escalator-related injury identification and prevention based on multi-module integrated system for public health*, Machine Vision and Applications, **33**, Art. 29, 2022.
10. Y. CAI, X. CHEN, C. ZHANG, K. LIN, X. WANG, H. LI, *Semantic scene completion via integrating instances and scene in-the-loop*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 324–333, 2021.
11. M. BORTNIKOV, A. KHAN, A. KHATTAK, M. AHMAD, *Accident recognition via 3D CNNs for automated traffic monitoring in smart cities*, Science and Information Conference, CVC 2019: Advances in Computer Vision, pp. 256–264, Springer, 2019.
12. F. WANG, J. QIAO, L. LI, Y. LIU, L. WEI, *Scene recognition of road traffic accident based on an improved faster R-CNN algorithm*, International Journal of Crashworthiness, 2021, DOI: 10.1080/13588265.2021.1959156.
13. T. LE, S. ONO, A. SUGIMOTO, H. KAWASAKI, *Attention R-CNN for accident detection*, 2020 IEEE Intelligent Vehicles Symposium (IV), pp. 313–320, 2020.
14. A. BRUNETTI, D. BUONGIORNO, G. TROTTA, V. BEVILACQUA, *Computer vision and deep learning techniques for pedes- trian detection and tracking: A survey*, Neurocomputing, **300**, pp. 17–33, 2018.
15. Y. CAI, B. LI, Z. JIAO, H. LI, X. ZENG, X. WANG, *Monocular 3d object detection with decoupled structured polygon estimation and height-guided depth estimation*, Proceedings of the AAAI Conference on Artificial Intelligence, **34**, pp. 10478–10485, 2020.
16. V. VISHNU, M. RAJALAKSHMI, R. NEDUNCHEZHIAN, *Intelligent traffic video surveillance and accident detection system with dynamic traffic signal control*, Cluster Computing, **21**, 1, pp. 135–147, 2018.
17. M. SONG, Z. HOU, *Model analysis of traffic emergency dispatching in intelligent transportation system under cloud computing*, International Journal of Internet Protocol Technology, **13**, *1*, pp. 46–54, 2020.
18. A. FINOGEEV, A. FINOGEEV, L. FIONOVA, A. LYAPIN, K. LYCHAGIN, *Intelligent monitoring system for smart road environment*, Journal of Industrial Information Integration, **15**, pp. 15–20, 2019.

19. M. DOUMA, *Automated video surveillance and machine learning: Leveraging existing infrastructure for cardiac arrest detection and emergency response activation*, Resuscitation, **126**, Art. e3, 2018.
20. K. HUANG, H. LEI, Z. JIAO, Z. ZHONG, *Recycling waste classification using vision transformer on portable device*, Sustainability, **13**, *21*, Art. 11572, 2021.
21. D. SARDDAR, S. ROY, P. SEN, *Edge multilevel edge server co-operation in content delivery network using hierarchical classification*, International Journal of Grid and Distributed Computing, **10**, *3*, pp. 41–52, 2017.
22. W. SHI, J. CAO, Q. ZHANG, Y. LI, L. XU, *Edge computing: Vision and challenges*, IEEE Internet of Things Journal, **3**, *5*, pp. 637–646, 2016.
23. B. KUMEDA, F. ZHANG, A. OLUWASANMI, F. OWUSU, M. ASSEFA, T. AMENU, *Vehicle accident and traffic classification using deep convolutional neural networks*, 2019 16th International Computer Conference on Wavelet Active Media Technology and Information Processing, IEEE, pp. 323–328, 2019.
24. K. HE, X. ZHANG, S. REN, J. SUN, *Deep residual learning for image recognition*, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778, 2016.
25. S. WOO, J. PARK, J. LEE, I. KWEON, *CBAM: Convolutional block attention module*, Proceedings of the European Conference on Computer Vision (ECCV), pp. 3–19, 2018.
26. Z. LIU, J. LI, Z. SHEN, G. HUANG, S. YAN, C. ZHANG, *Learning efficient convolutional networks through network slimming*, Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 2736–2744, 2017.

Received September 24, 2021