IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 20, 2020, accepted July 25, 2020, date of publication July 30, 2020, date of current version August 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3012995

A Robust Vehicle Detection Scheme for Intelligent Traffic Surveillance Systems in Smart Cities

ZHIYUAN WANG[1], JIFENG HUANG[1], NEAL N. XIONG[2], (Senior Member, IEEE), XIAOPING ZHOU[1], XIAO LIN[1], (Member, IEEE), AND THEODORE LEE WARD[2]

[1]College of Information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai 200234, China
[2]Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK 74464, USA

Corresponding authors: Jifeng Huang (jfhuang@shnu.edu.cn) and Xiaoping Zhou (zhouxp@shnu.edu.cn)

This work was supported by the Local Capacity Building Project of Shanghai under Grant 19070502900.

ABSTRACT Accurately obtaining road vehicle information is important in intelligent traffic surveillance systems for smart cities. Especially smart vehicle detection is recognized as the critical research issue of intelligent traffic surveillance systems. In this paper, a robust real-time vehicle detection method for the system is proposed. The method combines background subtraction model MOG2(Mixture of Gaussians) with a modified SqueezeNet model (H-SqueezeNet). The MOG2 model is utilized to create scale-insensitive Region of Interest (RoIs) from video frames. H-SqueezeNet is then proposed to accurately identify vehicle category. The effectiveness of the method was verified in CDnet2014 dataset, UA-DETRAC dataset and video data from a traffic intersection in Suzhou, China. The experiment results show that the method can achieves excellent detection accuracy in traffic surveillance systems, and achieve an average detection speed of 39.1 FPS.

INDEX TERMS Vehicle detection, H-SqueezeNet, MOG2, intelligent traffic surveillance system, smart city.

I. INTRODUCTION

In the past decades, surveillance cameras have spread across the traffic system for traffic control [1], [2], safe driving and abnormal detection [3], [4]. And vehicle detection application is more and more important for smart cities [5], [6]. Although cameras have been deployed on most urban roads, humans cannot review every monitor at the same time, and humans are unable to focus on monitors all the time. Therefore, a friendly intelligent traffic surveillance system is needed to help humans achieve intelligent traffic management [6], [7]. The first task of intelligent traffic surveillance is accurate vehicle detection [6]. It is the key technique in the most of traffic applications, such as road real-time monitoring, intelligent tracking and intelligent traffic control [1], [5]. Thus, the goal of our system is to achieve vehicle detection and category identification, while meeting real-time requirements.

During the past decade, some challenging vehicle detection benchmarks have been proposed for evaluation of various detection methods. Meanwhile, deep learning-based methods have achieved amazing achievements on vehicle detection and object detection fields, and can be divided into one-stage and two-stage detection algorithms. R-CNN [8], Fast R-CNN [9], Faster R-CNN [10] are the representative methods in two-stage field. These algorithms usually have high accuracy, but have huge computational complexity and are difficult to meet real-time performance. SSD [11], YoLoV1 [12], YoLoV2 [13] and YoLoV3 [14] are the representative methods in one-stage field, and have been made a trade-off between speed and accuracy. Usually, deep learning-based algorithms have complex architecture and computation. Meanwhile, scale-sensitivity also plagues the above methods. To deal with scale sensitivity, YoloV3 adopts anchor boxes, FPN [15], and uses three prediction branches to deal with different object size. CMNet [16] utilizes four prediction branches. Kim *et al.* [17] expanded the prediction branches of YoLoV3 to five and add SPP [18] to further alleviate scale-sensitivity. It can be concluded a ''rigorous conclusion'' that if we want stronger scale-insensitivity, add more prediction branches and more tricks. In object detection field, the above advanced methods and improvements are progressive and successful for the diversity of backgrounds and scenarios. However, the background and scenarios are relatively stable in the field of vehicle detection, especially in

The associate editor coordinating the review of this manuscript and approving it for publication was Sudipta Roy.

the field of video surveillance for smart cities. At this time, can we return to simplified thinking and design a powerful method to complete the detection? The paper utilizes it as the outset and designs a simple but powerful vehicle detection method for smart cities.

In this paper, we present a robust vehicle detection method, for fast vehicle detection in smart city surveillance. The method architecture is shown in Fig. 1. The method introduces robust MOG2 [19], [20] from the foreground extraction field to the regional proposal field, to enhance the scale-insensitivity. Then, the proposed method also designs H-SqueezeNet to increase the robustness in the term of accuracy. Usually, original MOG2 is used to extract vehicle foreground in the early urban surveillance system due to its robustness, we incorporate Suzuki's theory [21] and a series of morphological operations into it, and successfully utilize it to generate RoIs. Original SqueezeNet is an advanced lightweight network, we modified it into H-SqueezeNet, for better meet our needs. The H-SqueezeNet architecture is shown in Fig. 3.

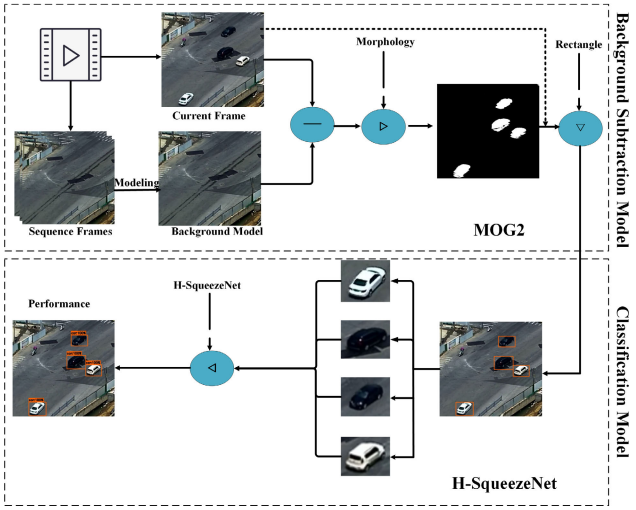

FIGURE 1. Architecture of the proposed method. The method extracts RoIs from video frame using MOG2, and then utilizes H-SqueezeNet model to identify vehicle category.

In summary, the main contributions of this paper are as follows:

1). A simple but powerful vehicle detection method is proposed for intelligent traffic surveillance systems. The architecture of the method is shown in Fig. 1.

2). The proposed method achieves scale-insensitivity by introducing MOG2, and utilizes H-SqueezeNet to ensure excellent performance. Meanwhile, the method can achieve real-time detection for traffic dataset by camera sensors from urban intersection, with minimal storage resources, making it more readily applied to the systems.

3). A modified SqueezeNet model (H-SqueezeNet) is proposed, which just retains the top four Fire modules in the original SqueezeNet and concatenates the last two Fire

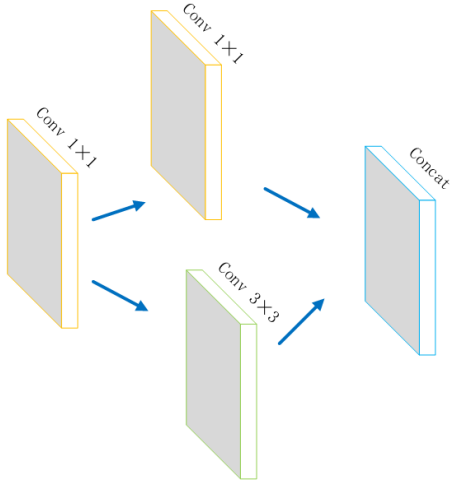

FIGURE 2. Architecture of Fire module. Fire module is utilized to construct H-SqueezeNet model.

module outputs. It achieves excellent performance in terms of accuracy. The model architecture is shown in Fig. 3.

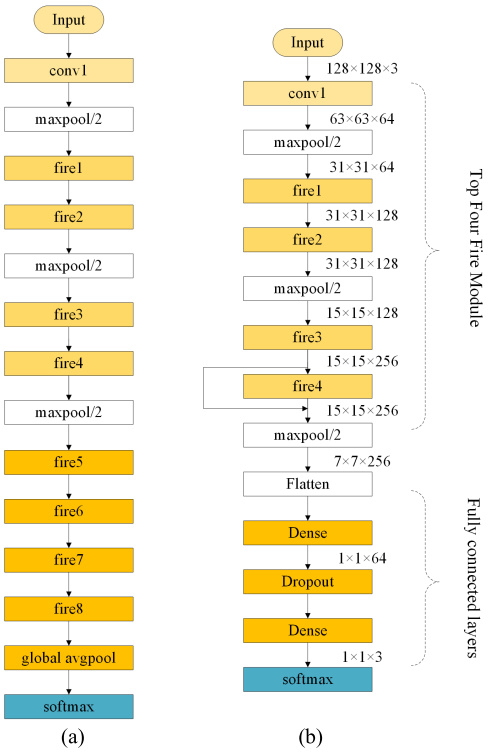

FIGURE 3. Model Architecture and comparison: (a) Architecture of SqueezeNet and (b) Architecture of H-SqueezeNet. The Fire module is designed in Fig.2.

The rest of the paper is organized as follows. Section 2 makes a brief overview about vehicle detection in traffic surveillance. Section 3 introduces a brief overview of the background subtraction model MOG2, which our method utilizes to extract RoIs from video frames. H-SqueezeNet is also proposed in Section 3. Section 4 presents the results and

experiments of the method. Finally, this paper is concluded in Section 5.

II. THE RELATED WORK

In this section, we make a brief introduction about vehicle detection in smart cities, especially in the field of urban surveillance.

At present, research on vehicle detection for intelligent traffic surveillance systems is mainly divided into two categories: traditional algorithms and methods based on deep learning. The first category mainly utilizes background subtraction, optical flow and descriptor-based methods. Vargas *et al.* [22] introduced a novel background subtraction method based on the sigma-delta filter, which is intended to be used in urban intelligent traffic scenes. This method introduced a confidence measurement for each pixel, making it get a more stable background for vehicle detection. Yan *et al.* [23] utilized two Histogram of Oriented Gradient (HOG) descriptors that generate features with lower dimensions and contain more vertical and horizontal gradient features, and then they applied AdaBoost classifier to finish vehicle detection. Wahyono and Jo [4] implemented illegally parked vehicle detection, as another important part in intelligent traffic surveillance systems. They utilized cumulative dual foreground differences and temporal event analysis in their work and successfully detected most illegally parked vehicles. Zhou *et al.* [24] proposed a novel scheme to carry out adaptive background estimation. They divided the input image into small non-overlapped blocks, RoIs of vehicle is then extracted from these blocks. Next, they utilized histogram for feature extraction and used PCA to get low-dimensional features. Finally, SVM was trained by extracted features to verify vehicle category. Kul *et al.* [25] designed a system for traffic control, they first utilized background subtraction method to extract the vehicle foregrounds. Then, they extracted geometry-based feature and utilize PCA to reduce feature dimension. Finally, three different classifiers were selected for three vehicle categories. Wang *et al.* [26] introduced an improved spatio-temporal sample consensus (ISTSC). They first detected moving vehicle through spatio-temporal sample consensus algorithm. Then, feature fusion methods are utilized to identify category in their work. However, the above algorithms are difficult to implement real-time detection or multi-class detection. Therefore, one important issue that needs to be solved of our method is to achieve real-time multi-class detection. More detailed traditional vehicle detection challenges and methods can be learned in [1], Buch *et al.* made a comprehensive review in this area, and an outlook for future research directions.

The second category mainly based on deep learning. Inspired by the great success of deep learning in the object detection field, some researchers are beginning to attempt to introduce deep learning to vehicle detection field. Wei *et al.* [27] tried to detect anomalous vehicles in traffic surveillance. They first employed background subtraction model to remove identified moving vehicles, and then utilized Faster R-CNN [10] to detect remaining anomalous vehicles. Mhalla *et al.* [7] introduced a modified Faster R-CNN (MF R-CNN) for vehicle detection and pedestrian detection, they removed the fourth Max-Pooling layer, and replaced the remaining Max-Pooling layers by Stochastic-Pooling layers in original Faster R-CNN. Finally, they made an embedded system for intelligent traffic surveillance. Zhang *et al.* [16] proposed a connect-and-merge convolutional neural network (CMNet) for fast vehicle detection in urban traffic surveillance. CMNet contains a connect-and-merge residual network (CMRN) and a multi-scale prediction network (MSPN). CMRN is utilized to extract features and MSPR is utilized to finish detection. Hu *et al.* [28] found CNN-based algorithms have scale-sensitive problem. However, it is common that vehicles have a large variance of scales in traffic surveillance. Hence, they proposed a Scale-Insensitive Convolutional Neural Network (SINet) to address the issue. In SINet they introduced a context-aware RoI pooling to retain contextual information of small-scale objects and then proposed a multi-branch decision network to finish detection. Kim *et al.* [17] introduced spatial pyramid pooling (SPP) into YoLoV3 and add more prediction layers in it, making it better fit to multi-scale vehicle detection in traffic surveillance and alleviate scale-sensitive problems. Another effort to mitigate scale-sensitive problems, especially for small vehicle target, Hong *et al.* [29] modified YoLoV3 to a new pyramid structure based on codec module, which can achieve good performance in actual vehicle detection demand. Sentas *et al.* [30] tried to introduce Tiny-YoLo into real-time vehicle detection field, and built TPSdataset for test. Zhou *et al.* [31] proposed a fast vehicle detection algorithm DAVE for urban traffic surveillance. DAVE contains a fast vehicle proposal network (FVPN) and an attribute learning network (ALN). FVPN is designed to find vehicle location, and then ALN is utilized to verify vehicle category and other information of the vehicle. It is clearly that these methods based on deep learning are mainly stuck in scale-sensitive issues.

In summary, researchers found that traditional algorithms are robust, but they are unable to handle real-time multi-class detection and have insufficient accuracy. Deep learning-based algorithms perform well in speed and accuracy, but they exhibit scale-sensitive issues, making them have some deficiencies to deal with multi-scale vehicle detection in real time traffic surveillance. Hence, if we can utilize traditional algorithm MOG2 [19], [20] to create robust scale-insensitive RoIs and exploit deep learning model H-SqueezeNet to finish detection, the advantages of both methods will be inherited and shortcomings of each method will be avoided. In our work, we didn't combine MOG2 with VGG, or imitate Wei *et al.* [27] combine MOG2 with Faster R-CNN. Because we want to build a simple but powerful method. If the above methods are adopted, they can still get good detection performance, but they will cause some problems, such as model redundancy, high model complexity and low running speed.

III. OUR PROPOSED VEHICLE DETECTION METHOD

The architecture of our method is illustrated in Fig.1, this method combines traditional background subtraction model MOG2 [19], [20] with lightweight deep learning model H-SqueezeNet. First, MOG2 is utilized to extract RoIs from video frames for the vehicle detection task, this is an efficient algorithm using Gaussian mixture probability density. After that, H-SqueezeNet will be trained to verify vehicle category, which just retains the top four Fire modules from the original SqueezeNet and concatenates the last two Fire module outputs. The detailed architecture of H-SqueezeNet model is shown in Fig. 3. The following sections will explain the proposed method in detail.

A. BACKGROUND MODELING AND SUBTRACTION

During vehicle detection, the first stage is to extract RoIs. In the past, researchers have usually utilized sliding window, selective search or RPN to carry out regional proposal task. However, MOG2 [19], [20] is used to generate RoIs in this paper. Because MOG2 model is scale-insensitive, it can avoid scale-sensitive problems by directly using deep learning model to generate robust RoIs. In fact, the original MOG2 model is used to extract foregrounds, not RoIs. We utilized [21] and accompanied many morphological operations to make it possible to extract RoIs. The detailed stage of MOG2 to extract RoIs is shown in Algorithm 1.

From Fig. 1, it can be found that a background model needs to be built in background subtraction model. A good background model is required to extract better foregrounds. Fortunately, one important characteristic of MOG2 is that it selects an appropriate value for the Gaussian mixture model (GMM) for each pixel in the image to create a robust background model, making it better for adapting varying scenes. In our experiment, the number of GMM component is set to five. For low memory and fast speed, it also can change the GMM number to three. The background model [19], [20] can be approximately represented as:

$$p\left(\bar{x} \mid \chi_T, BG\right) \sim \sum_{m=1}^{M} \hat{\pi}_m N\left(\bar{x}; \hat{x}_m, \delta_m^2 I\right), \tag{1}$$

where M is the GMM number and I is an Identity Matrix. $\hat{x}_m$ is the estimate of the mean and δ_m is the estimate of the variance. $\hat{\pi}_m$ represents the weight of single GMM and accords with normal distribution. $\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha(O_m^t - \hat{\pi}_m)$. Parameter α is the background update rate. The more details of MOG2 can be learned in [19], [20].

In MOG2, the background model is not static, it needs to be updated according to the background update rate α, where the $\alpha = 1/T$. T is a hyperparameter, representing the time period. Usually, one larger T value allows MOG2 to get better background, but with more time consumption. One smaller T value can lighten time consumption, but with poor performance. Hence, a reasonable time period T needs to be selected to better update GMM parameters. Meanwhile, a larger T value allows MOG2 to better detect vehicles and relieves RoIs fluctuation problems of large vehicles. In our experiment, we recommend that the value of $T >= 700$ *frames*. The effect of the T value will be evaluated in the experiment.

When we get the background model b_g, foreground f_g can be extracted through subtraction:

$$f_g = C_f - b_g, \tag{2}$$

where C_f is current frame. When we get the foreground f_g, the RoIs can be obtained through the procedure in Algorithm 1. Next, an applied detail of MOG2 will be introduced.

In this paper, MOG2 simultaneously detects vehicle and vehicle shadow [33], which prevents the detector view the vehicle shadow as a vehicle. Though vehicle shadow detection will increase running time, higher quality regional proposals can be obtained. Moreover, vehicle shadow detection relieves the RoIs adhesion problem, while mitigating the problem of the detector treating two adjacent cars as same car. The detailed performance improvement will be shown in the experimental section.

Algorithm 1 Extract RoIs Through MOG2

Input: Video frames.
Output: RoIs.
Description:
img represents the video frame; Img_i is the pixel in *Img*; a_n is the area of x_n; $\{x_1 x_2 \cdots x_n\}$ are the contours in *Img*. *Th* is threshold calculated by OTSU [32] method. *AreaTh* is threshold to filter valid RoIs.
Procedure:
1. Initialize MOG2.
2. while true do:
3. Foreground extraction from the *img,* the processed *img* is recorded as *Img*.
4. Image thresholding through OTSU thresholding method.
$$Img_i = \begin{cases} 0 & Img_i < Th; \\ 255 & Img_i \geq Th; \end{cases}$$
5. Morphological process.
6. Find Contours from *Img*. Marked as $\{x_1 x_2 \cdots x_n\}$
7. for $j < n$
8. Calculate contour area of x_j. Marked as a_j.
9. if $a_j \geq AreaTh$:
10. Annotate x_j with Rectangle Box. Modify x_j into RoIs.
11. Waiting enter to H-SqueezeNet.
12. else:
13. Continue.
14. end if
15. end for
16. end while

B. FINE-TUNE H-SQUEEZENET

In this paper, H-SqueezeNet is also proposed, which just retains the top four Fire modules in the original SqueezeNet. Next, we concatenate the last two Fire module outputs to increase the available information in the feature map. Then, two fully connected (FC) layers will be added, employing the softmax classifier to complete classification task. Fig. 3 shows the architectural details of the H-SqueezeNet model. Compared to the original SqueezeNet, H-SqueezeNet has a smaller model size. Meanwhile, we give the main architecture of H-SqueezeNet with layers and parameters in Table 1.

TABLE 1. The main architectural dimensions of H-SqueezeNet.

Layer	Filter	Channel	Stride	Output size
Input				128×128
Convolutional	3×3	64	2×2	63×63
MaxPool	3×3		2×2	31×31
Fire module	1×1/3×3	16/64	1×1	31×31
Fire module	1×1/3×3	16/64	1×1	31×31
MaxPool	3×3		2×2	15×15
Fire module	1×1/3×3	32/128	1×1	15×15
Fire module	1×1/3×3	32/128	1×1	15×15
Residual				15×15
MaxPool	3×3		2×2	7×7

At training time, pre-trained weights on the ImageNet will be utilized to initialize model. Then, H-SqueezeNet will be fine-tuned by Adaptive Moment Estimation (Adam) with the traffic data from a traffic intersection in Suzhou, China. The training stage of H-SqueezeNet is shown in Algorithm 2.

As for the activation function, the original settings in SqueezeNet are retained, using the rectified linear unit function (ReLU) in the Fire modules. Then, the leaky ReLU function will follow the FC layers. Leaky ReLU is a modified version of ReLU, where the function output has a small slope for negative input. Since the derivative is always non-zero, this can reduce the appearance of silent neurons, solving the problem that ReLU does not learn after encountering negative intervals. Leaky ReLU is defined as:

$$\phi(\mathrm{x}) = \begin{cases} x, & if\ x > 0; \\ 0.1x, & if\ x < 0. \end{cases} \tag{3}$$

During training, the categorical cross entropy loss function will be utilized to optimize our model:

$$loss = -\sum_{i=1}^{n} \hat{y}_{i1} logy_{i1} + \hat{y}_{i2} logy_{i2} + \cdots + \hat{y}_{im} logy_{im}, \tag{4}$$

where n and m represent the number of samples and the number of categories, respectively. $\hat{y}$ represents the true value and y represents the prediction value.

In actual operation, we need to make the loss function pay more attention to the categories which have fewer samples, alleviating the problem of sample imbalance. In this paper, we add loss factors to the loss function for different vehicle categories to make the model train smoothly and prevent it overfitting. The modified loss function is defined as follows:

$$loss = -\sum_{i=1}^{n} \lambda_1 \hat{y}_{i1} logy_{i1} + \lambda_2 \hat{y}_{i2} logy_{i2} + \cdots + \lambda_m \hat{y}_{im} logy_{im}. \tag{5}$$

From Table 2, loss factor λ values have been listed for different vehicle categories. λ is calculated as:

$$\lambda_i = \frac{C_n}{nN_i}, \tag{6}$$

where C_n represents the total number of samples. N_i represents the sample amount of class i. n is the number of vehicle categories.

TABLE 2. λ of different vehicle category.

Category	Value of λ
Bus	0.90
Car	0.68
Truck	2.36

In the end, in order to improve the detection efficiency, we didn't select the traditional SVM classifier to finish the classification task, but selected softmax classifier, a multi-class classifier:

$$p_i = \frac{\alpha^i}{\sum_j \alpha^j}. \tag{7}$$

Algorithm 2 Training Stage of H-SqueezeNet

Input: Training dataset.
Output: Model weights w_t of H-SqueezeNet
Description:
λ_i is the loss factor of different categories; n is the number of categories; $\{y^{(1)}, y^{(2)}, \ldots, y^{(m)}\}$ is the dataset; m_t is the velocity and n_t is the learning rate; w_t is the parameter of model weight. C_n is the total number of samples. N_i represents the sample amount of class i.
Training:
1. Apply the model on the ImageNet dataset for pre-training.
2. Calculate the loss factor λ_i of different vehicle categories.

$$\lambda_i = \frac{C_n}{nN_i}$$

3. for: t<k
4. Calculate gradient: $g = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial L(y^{(i)}, f(x^i; w))}{\partial w}$
5. Update velocity: $m_t = \mu * m_{t-1} + (1-\mu) * g_t$
6. Update learning rate: $n_t = \nu * n_{t-1} + (1-\nu) * g_t^2$
7. Correction m_t: $\hat{m}_t = \frac{m_t}{1-\mu^t}$
8. Correction n_t: $\hat{n}_t = \frac{n_t}{1-\nu^t}$
9. Update parameter: $w_t = w_{t-1} - \frac{\hat{m}_t}{\sqrt{\hat{n}_t + \epsilon}} * \eta$
10. end for

In the above formula, α is a vector, and α^i represents one of the values in vector α. The above formula can map the values in vector α to the (0, 1) interval, using p_i to represent the final probability.

C. ABLATION STUDY

1) H-SQUEEZENET ARCHITECTURE STUDY

After extracting RoIs from the video frames, the next important stage is classification. The reason why we remove redundant fire modules and retain top four Fire modules in this paper is because we found that the feature map size unchanged after the top four fire modules. The feature map maintains size (7, 7, n), and only the channels change. The paper utilizes it as breakthrough to simplify model. Because we just retain the top four Fire modules, the information of feature map would be reduced. As compensation, the last two Fire module outputs have been concatenated in the experiment. The detailed architecture of H-SqueezeNet can be found in Fig. 3. And from Fig. 4, it can be readily found that the highlighted areas between the two models are similar, meaning that the feature map of H-SqueezeNet already contains enough context information to complete its task.

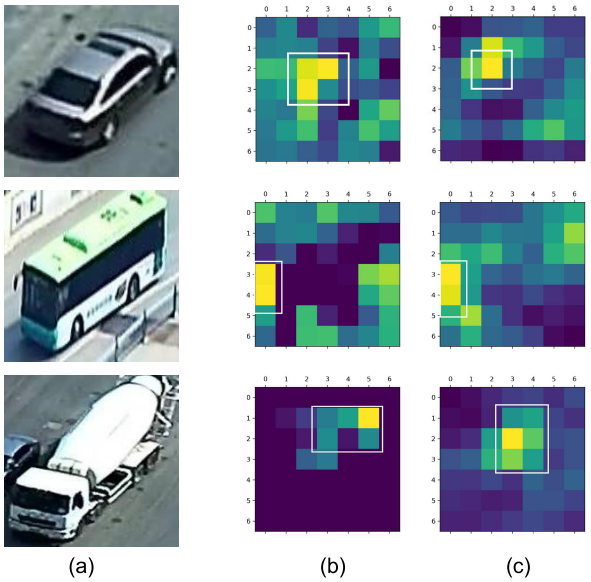

FIGURE 4. Feature map comparison. (a) Original image. (b) Feature map of H-SqueezeNet. (c) Feature map of SqueezeNet.

As for why to choose H-SqueezeNet model, it is well known that there are many excellent models, such as AlexNet [40], VGG [41], Inception_v3 [42] and ResNet [43]. But such models usually have too many model parameters and heavy calculation, which is insufficient to meet real-time. Therefore, some lightweight model architectures such as SqueezeNet [35], MobileNet [36], ShuffleNet [37] and Xception [38] were born. In this paper, the SqueezeNet was selected firstly in the experiments. In order to further reduce the model size, it was modified and the H-SqueezeNet model was proposed.

2) METHOD SELECTION DISCUSSION

In recent years, some excellent deep learning detection algorithms such as YoLoV3 [14] have appeared. They are not directly applied to the field of intelligent traffic surveillance systems and other smart city application, because these detectors are universal detectors, not domain-specific detectors. For example, YoLoV3 can achieve excellent performance in normal scene, but its property may be ''limited'' when it is applied to a specific field. In other words, it perfect, but may not meet the specific needs in specific fields well. Next, we will have a brief discussion and explanation.

Through Fig. 5, it can be found that YoLoV3 has excellent performance for traffic surveillance data, but commonly produces false positives (FP). Clearly, FP need to be reduced in intelligent traffic surveillance systems. Hence, this work makes a trade-off, choosing a slower method to generate RoIs, in order to get robust regional proposals and lower FP. Specifically, the paper utilizes MOG2 for scale-insensitive RoIs and lower FP, while introducing H-SqueezeNet to achieve excellent accuracy, taking full advantages of both methods, and avoiding the shortcomings of the two methods as much as possible.

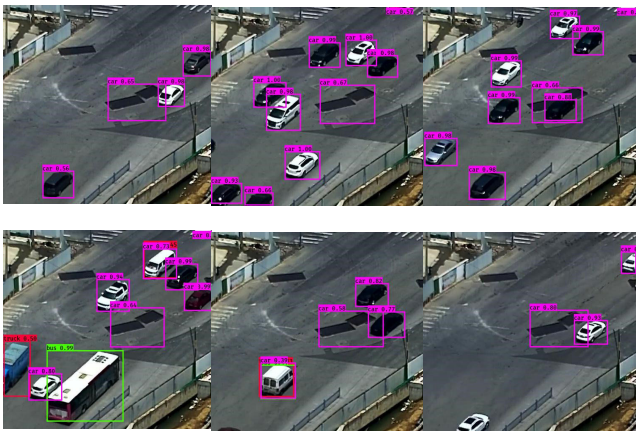

FIGURE 5. Performance of YoLoV3 in traffic surveillance.

In our experiment, we also utilized SSD [11], YoLoV2 [13], RetinaNet [39] and Faster R-CNN [10] to carry out vehicle detection experiments, and compared these state-of-the-art methods with our proposed method. More specific comparisons and discussions will be demonstrated in the following section.

IV. PERFORMANCE ANALYSIS AND EVALUATION

In this section, the performance of the proposed method will be evaluated. First, we evaluate the method on traffic data from a traffic intersection in Suzhou, China. Then, CDnet2014 [44], Highway videos from [45] and UA-DETRAC [46] dataset will be utilized to prove its generalization and robustness. Moreover, the performances of MOG2 and H-SqueezeNet are also introduced and analyzed below, respectively.

A. DATASET AND EXPERIMENTAL CONFIGURATION

1) DATASET DESCRIPTION

We first utilize our traffic dataset collected from a traffic intersection in Suzhou, China, which contain a series of video sequences for training and testing. These videos are taken from different times in the daytime, i.e., these videos have a variety of illumination and shadow conditions. Moreover, the dataset has different traffic flow conditions such as light traffic, normal traffic and traffic jam. In summary, the dataset has complex conditions that need to be addressed, which makes the work more challenging. The following experiments demonstrate that our method is robust to these conditions. In addition, further experiments on public datasets will be listed to further demonstrate the robustness of the method.

As further evidence, we use CDnet2014 dataset [44] which contains 22 additional videos spanning 5 categories more than CDnet2012 to evaluate our method. CDnet2014 contains 53 video sequences representing various challenges divided into 11 categories such as Baseline, Camera Jitter, Dynamic Background, Shadow, Intermittent Object Motion, Thermal, Bad Weather, Low Framerate, Night Video, PTZ and Turbulence. In our experiment, video sequences will be picked from different categories. On the other hand, the videos in CDnet2014 dataset have a low-resolution and varies from 320×240 to 720×486, which further increase the difficulty of the vehicle detection task. For greater credibility, we also utilize the proposed method to perform on Highway videos from [45].

Then, we utilize UA-DETRAC [46] dataset to further test our method. The UA-DETRAC dataset consists of 100 video sequences. These videos represent various conditions including urban highway and traffic crossings. In addition, the dataset has four weather conditions (i.e., rainy, night, sunny, and cloudy). Meanwhile, the dataset is recorded by Canon EOS 550D camera at 25 FPS with 960×540 resolution at Beijing and Tianjin in China. The more details can be learned in [46].

2) MOG2 MODEL CONFIGURATION

For multi-class vehicle detection, the first stage is to extract RoIs. MOG2 is utilized to finish it in this paper. The resolution of input video is recommended to be from 320×240 to 1080×720. The number for GMM is set to five. The coding platform is Opencv for python. The variance threshold is set to 16, which is used to decide if the sample is well described by the background model or not. The value of initial variance of each gaussian component is 15.

In order to run faster, cuda acceleration technology can be introduced. Because our current work has met the real-time requirements, this technology is not used now. In the future, if higher speed requirements arise, we will use cuda technology to accelerate MOG2, and port it to the C ++ platform.

3) H-SQUEEZENET MODEL TRAINING DETAILS AND EXPERIMENTAL CONFIGURATION

For multi-class vehicle detection, the second stage is to identify the vehicle category. In this paper, H-SqueezeNet is selected to accomplish this. The pre-trained weights on ImageNet are utilized to initialize H-SqueezeNet model. Then, traffic data from traffic intersection are utilized to fine tune the model. In detail, the H-SqueezeNet model with 128×128 resolution input, is trained end-to-end by Adaptive Moment Estimation (Adam), where the learning rate is 0.001 and batch size is 64. At the same time, in order to enhance model robustness, a real-time data augmentation strategy will be adopted in model fine-tune process. Table 3 shows the detailed data augmentation strategies during the model training process.

TABLE 3. Data augmentation strategy and parameter value in training time.

Strategy	Value
Zoom	0.2
Shear	0.2
Width shift	0.2
Height shift	0.2
Rescale	True
Horizontal flip	True
Fill mode	Nearest

Meanwhile, the experiments and comparison experiments need to perform in the same environment configurations. Detailed environment configurations are listed in Table 4. Keras platform will be utilized in the experiment, which is an efficient and powerful framework for deep learning.

TABLE 4. Experimental configuration details.

Item	Configuration
CPU	Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
GPU	NVIDIA GeForce GTX TITAN XP
Memory	32GB
Cuda	CUDA 9.2
Cudnn	CUDNN 7.5
Python	Python 3.6
Opencv	Opencv 3.4

4) EVALUATION PROTOCOL

For MOG2, FPS will be listed in the experiment. As for the H-SqueezeNet model, accuracy, model size, precision P, recall R, and F-measures were utilized as evaluation protocols. F-measures is defined as:

$$F = \frac{2 \times P \times R}{P + R}. \tag{8}$$

As for the most important detection results, the detection performances and comparison experiments will be displayed directly in the experimental section.

B. EXPERIMENTAL RESULTS AND ANALYSIS

In this part, the performance of the proposed method will be demonstrated. And we compared the detection performance of our method with advanced methods such as YoLoV2, YoLoV3, RetinaNet, SSD, Faster R-CNN. Moreover, the performances of MOG2 and H-SqueezeNet are also introduced. Meanwhile, we compared MOG2 with other background subtraction methods, the paper also compared H-SqueezeNet with some state-of-the-art models.

1) PERFORMANCE OF MOG2

In this paper, MOG2 is utilized to extract foregrounds and RoIs from video frames. To prove its effectiveness, MOG2 is compared with various background subtraction models such as CNT, GSOC, GMG and LSBP in Table 5, it can be found that MOG2 performs well in most of background subtraction models. Even with the additional time consumption of vehicle shadow detection, MOG2 is excellent at running speed. Compared with other methods, MOG2 is more flexible and adjustable, and other methods are generally unable to perform vehicle shadow detection.

TABLE 5. Comparison between background subtraction models. This experiment was performed on I5-8250U. The final run will be performed on server.

Method	Fps	Vehicle Shadow Detection
Background subtraction KNN	24.55	T
Background subtraction KNN	38.50	F
Background subtraction CNT	30.99	F
Background subtraction GMG [47]	10.31	F
Background subtraction GSOC	13.55	F
Background subtraction LSBP [48]	9.10	F
Background subtraction MOG [49]	24.08	T
Background subtraction MOG2	45.10	F
Background subtraction MOG2	29.35	T

To achieve better performance, MOG2 is allowed to carry out shadow detection in the experiment, which can prevent the detector from misjudging adjacent vehicles as the same vehicle. Fig. 6 shows the resulting improvement through vehicle shadow detection. And time period T is 700 *frames* in the experiment. By the formula $a = 1/T$, it can be easily seen that the background update rate $a = 1/700$. The experiment result in Fig. 7 shows that the T value can make MOG2 performs well and relieves the RoIs area fluctuation problem of large vehicles successfully.

2) PERFORMANCE OF H-SQUEEZENET

In the field of deep learning, learning curve is an important evaluation criterion for evaluating network models. The loss learning curve and acc learning curve of H-SqueezeNet model are exhibited in Fig. 8. As can be clearly seen in Fig. 8, training loss and validation loss are reduced with the growth of epochs, which proves that H-SqueezeNet model did not exhibit "over-fit" or "under-fit" phenomenon during the training stage. Meanwhile, Fig. 9 shows the confusion

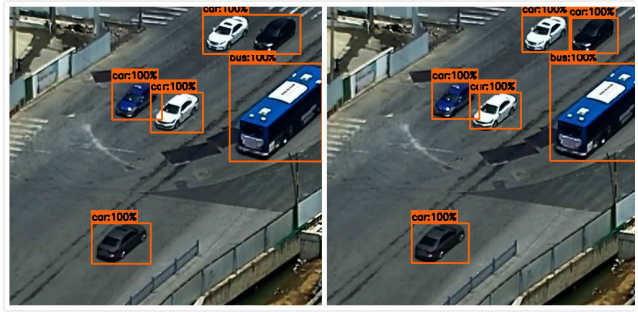

FIGURE 6. Performance improvement through vehicle shadow detection. Left: Performance of not detecting vehicle shadow. Right: Performance of detecting vehicle shadow.

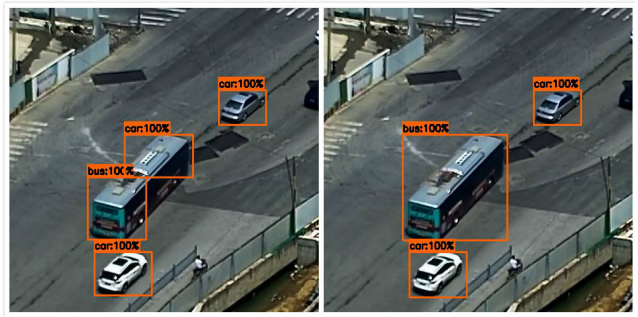

FIGURE 7. Performance comparison of different T values. Left: T value below 700 frames. Right: T = 700 frames.

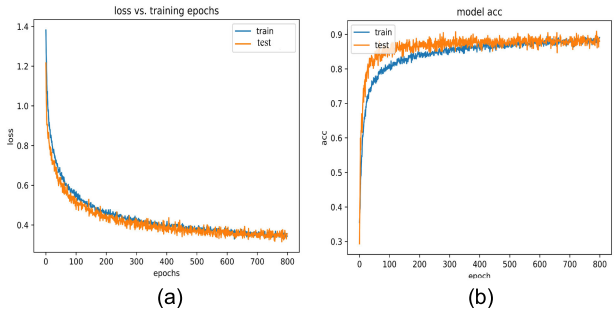

FIGURE 8. Learning curves of H-SqueezeNet model. (a) Loss learning curve of H-SqueezeNet. (b) Acc learning curve of H-SqueezeNet.

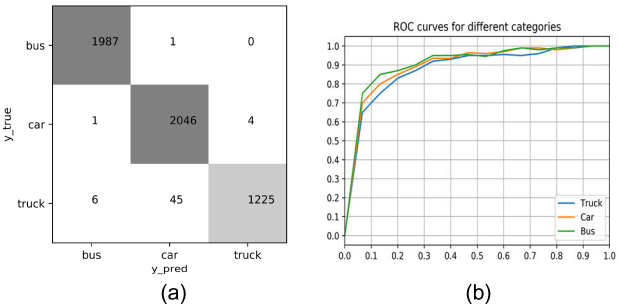

FIGURE 9. The confusion matrix and ROC curves of H-SqueezeNet model. (a) Confusion matrix of model. (b) ROC curves of model.

matrix and ROC curves of H-SqueezeNet. Through the confusion matrix, it can be easily seen that accuracy of the H-SqueezeNet is 98.93%. Moreover, it can be seen from the

confusion matrix, the truck classification efficiency is obviously lower than other vehicle categories, of which 6 trucks are divided into buses, 45 trucks are misjudged as cars. We speculate that such a phenomenon is caused by truck category diversity and fewer truck data samples. In urban traffic, some intersections have strict regulations for trucks. If intelligent traffic surveillance systems can detect vehicles accurately, it can detect illegal trucks in time. Hence, improving the accuracy of truck identification is one of our next goals.

As mentioned above, H-SqueezeNet utilizes a real-time data augmentation strategy in the training stage. However, it is worth noting that after the training stage is completed, only the normalization operation is retained during the validation stage to ensure credibility. Moreover, the evaluation report of H-SqueezeNet is presented in Table 6. As can been seen from Table 6, H-SqueezeNet achieves excellent scores for different vehicle categories.

TABLE 6. Performance of H-SqueezeNet for different vehicle categories.

	Precision	Recall	f1-score
Bus	99.65%	99.95%	99.80%
Car	97.80%	99.76%	98.77%
Truck	99.67%	96.00%	97.80%

The performance of H-SqueezeNet is shown above. Here, H-SqueezeNet model is compared with original SqueezeNet [35] and other state-of-the-art networks such as VGG16 [41], VGG19, Inception_v3 [42], ResNet [43] and darknet_53 [14] in the Table 7. In H-SqueezeNet comparative experiments, in order to maintain credibility, this paper utilized the same training dataset, testing dataset, hardware configuration and hyper-parameters for every model. It can be concluded clearly from Table 7 that most of state-of-the-art models can achieve more than 75% accuracy, especially H-SqueezeNet can achieve 98.93% accuracy. Meanwhile, H-SqueezeNet only needs 3.56 MB of storage, making it easier to deploy to intelligent traffic surveillance systems in smart cities. Moreover, H-SqueezeNet only needs 10 *ms* to finish the classification stage, which means it can reach real-time performance in the systems.

TABLE 7. Comparison between our proposed model and other state-of-the-art models.

Model	Model Size (MB)	Accuracy	Precision	Recall	f1-score
SqueezeNet	9.28	95.71%	95.50%	95.01%	95.19%
VGG16	112.51	89.71%	90.59%	87.50%	88.30%
VGG19	156.03	92.27%	94.01%	90.22%	91.32%
Inception_v3	103.78	87.50%	59.05%	63.54%	60.73%
ResNet v1	200.84	82.19%	91.23%	89.58%	88.78%
ResNet v2	97.76	75.00%	83.33%	83.33%	77.78%
darknet_53	158.89	90.63%	92.08%	93.16%	92.54%
H-SqueezeNet	3.56	98.93%	99.04%	98.75%	98.79%

In the above experiment, some state-of-the-art models have been compared with H-SqueezeNet from five aspects:

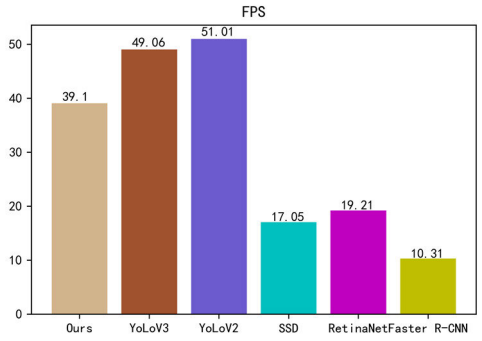

FIGURE 10. The avg FPS comparison with other advanced methods.

accuracy, precision, recall, f1-score and model size. But some models such as Inception_v3 underperform in terms of precision, recall and f1-score, we speculate that the input resolution is too low, resulting in some convolution channels in the inception module of Incetion_v3 being silent. Because the channel is silent, the model is not learned well. It means that such models are difficult to handle the condition with lower RoIs size. But H-SqueezeNet is robust to this condition.

After comparing with a series of deep learning models. This paper also compares H-SqueezeNet with some traditional methods, such as HOG method, SIFT method, SURF method and LBP method. Detailed comparison results are listed in Table 8. Through Table 8, it can clearly draw conclusions that traditional methods are more sensitive to cars, but less effective for buses and trucks. In sharp contrast, our algorithm performs well for different vehicle categories, thanks to the algorithm based on deep learning model, which can learn global features well, unlike traditional algorithms, which focus on local features. At the same time, through using a convolutional network, the model can learn features autonomously without the need to select hand-crafted features by user.

TABLE 8. Precision of different vehicle categories by the proposed model and traditional methods.

	Bus	Car	Truck
HOG+SVM	72.37%	96.00%	89.66%
SIFT+SVM	76.32%	92.38%	72.41%
SURF+SVM	71.05%	93.33%	70.65%
LBP+SVM	73.68%	87.62%	82.76%
H-SqueezeNet	99.65%	97.80%	99.67%

3) COMPREHENSIVE PERFORMANCE

In this part, we don't list complicated tables and analysis, but directly list the intuitive performance in Fig. 11 and comparison in Fig. 12-14, continuing the simplified thinking.

To better meet the needs of smart cities, the vehicle detection algorithms in the urban traffic surveillance field should be as fast as possible. We list the FPS between different methods in Fig. 10. The YoLo family, as the most advanced algorithms, easily meets the speed requirements. Meanwhile, we utilize the feature of traffic surveillance in

FIGURE 11. The first row is the performance of the method in our own dataset from a traffic intersection in Suzhou, China. The second and third row are the performance of the method in CDnet2014 dataset [44] and Highway videos from [45]. The fourth row is the performance of the method with poor quality images in CDnet2014 dataset. the fifth and sixth row are the performance of the method in UA-DETRAC dataset [46].

smart cities, design a simple and fast algorithm, which can reach 39.1 FPS. Next, the method accuracy can be represented by H-SqueezeNet, we have listed the performance and comparison in Table 7 and Table 8.

Then, we evaluate the method detection performance on our own dataset collected from a traffic intersection in Suzhou, China. The performances have been shown in Fig. 11. We can easily draw conclusion from it that our method provides excellent performance for different traffic flows, and also obtain high detection accuracy.

Further research, CDnet2014 [44] and Highway videos from [45] are used to evaluate the method. It can be seen from Fig. 11 that the method can be utilized in varied traffic scenarios and angles, proving its robustness and generalization in different traffic scenarios. Meanwhile, it also can draw conclusions from Fig. 11 that the method can achieve high detection accuracy even for vehicles of different scales, and the method can adjust to different monitor angles. In other words, the method is scale-insensitive, which benefits from the deployment of MOG2. Meanwhile, we have selected some low quality videos with bad weather conditions from CDnet2014 to evaluate the method in Fig. 11. Then, UA-DETRAC [46] dataset is used to further test our method. We selected scenes from different angles and scenes to verify

FIGURE 12. Performance comparison in our own dataset. (a) Performance of YoLoV3. (b) Performance of RetinaNet. (c) Performance of SSD. (d) Performance of YoLoV2. (e) Performance of Faster R-CNN. (f) Performance of our method.

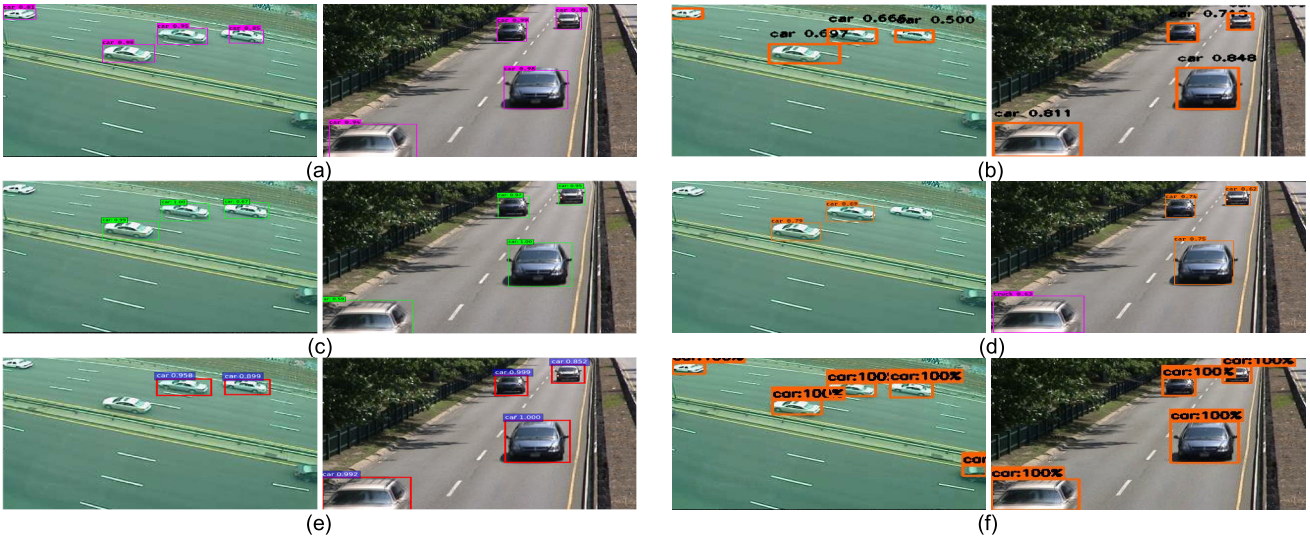

FIGURE 13. Performance comparison in CDnet2014 dataset. (a) Performance of YoLoV3. (b) Performance of RetinaNet. (c) Performance of SSD. (d) Performance of YoLoV2. (e) Performance of Faster R-CNN. (f) Performance of our method.

the robustness of the method. The detection performances are also shown in Fig. 11.

In the above experiment, the performance of our method has been demonstrated. Then, we will list the comparisons with state-of-the-art methods such as YoLoV2, YoLoV3, RetinaNet, SSD and Faster R-CNN. Fig. 12 shows the performances of these state-of-the-art detection frameworks and our method in our own dataset. It can be found from Fig. 12 that most methods perfrom well, but exist little of scale-sensitive issues or false positives (FP). To solve the above issues, this paper makes a trade-off. Our method choose a slower but robust method MOG2 to create RoIs and enhance scale-insensitivity. Then, H-SqueezeNet is utilized to guarantee accuracy. As mentioned above, H-SqueezeNet only needs

FIGURE 14. Performance comparison in UA-DETRAC dataset. (a) Performance of YoLoV3. (b) Performance of RetinaNet. (c) Performance of SSD. (d) Performance of YoLoV2. (e) Performance of Faster R-CNN. (f) Performance of our method.

10 *ms* to identify vehicle category. With the additional time consumption of MOG2, the integral method can reach an average speed of 39.1 FPS, which means that the method can achieve real-time performance in intelligent traffic surveillance systems of smart cities.

Meanwhile, we further compare our method with other state-of-the-art methods on CDnet2014 and UA-DETRAC datasets. Through comparison experiments on public datasets, further verify the performance of the method. The Fig. 13 and Fig. 14 show the comparison experiments on CDnet2014 and UA-DETRAC, respectively.

V. CONCLUSION AND FUTURE WORK

In this paper, a simple but powerful vehicle detection method has been proposed, for real-time multi-class vehicle detection of intelligent traffic surveillance systems in smart cities. The method utilizes MOG2 to create RoIs from video frames, making it avoid scale-sensitive problems caused through directly using deep learning to finish region proposal. Then, H-SqueezeNet is utilized to accurately identify vehicle category. Moreover, our method can meet real-time in the systems. In experiments, the proposed method achieves excellent performance on CDnet2014 [44], Highway videos from [45], UA-DETRAC [46] and video data from a traffic intersection in Suzhou, China. The performance of the method shows that it can be applied to intelligent traffic surveillance systems and other smart city applications.

In future work, we will strive to introduce more vehicle categories such as tractors, urban railcars. Then, we will study cuda technology to accelerate MOG2 for faster detection speed. In other fields, we will introduce license plate detection and traffic density analysis, and combine them with vehicle detection into a comprehensive system for smart cities.

ACKNOWLEDGEMENT

The authors would like to thank the reviewers for their work on this article.

REFERENCES

[1] N. Buch, S. A. Velastin, and J. Orwell, "A review of computer vision techniques for the analysis of urban traffic," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 920–939, Sep. 2011.
[2] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur, "A survey of vision-based traffic monitoring of road intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2681–2698, Oct. 2016.
[3] Z. Fang, F. Fei, Y. Fang, C. Lee, N. Xiong, L. Shu, and S. Chen, "Abnormal event detection in crowded scenes based on deep learning," *Multimedia Tools Appl.*, vol. 75, no. 22, pp. 14617–14639, Nov. 2016.
[4] Wahyono and K.-H. Jo, "Cumulative dual foreground differences for illegally parked vehicles detection," *IEEE Trans. Ind. Informat.*, vol. 13, no. 5, pp. 2464–2473, Oct. 2017.
[5] L. Hu and Q. Ni, "IoT-driven automated object detection algorithm for urban surveillance systems in smart cities," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 747–754, Apr. 2018.
[6] Y. Tang, C. Zhang, R. Gu, P. Li, and B. Yang, "Vehicle detection and recognition for intelligent traffic surveillance system," *Multimedia Tools Appl.*, vol. 76, no. 4, pp. 5817–5832, Feb. 2017.
[7] A. Mhalla, T. Chateau, S. Gazzah, and N. E. B. Amara, "An embedded computer-vision system for multi-object detection in traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 11, pp. 4006–4018, Nov. 2019.
[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, Dec. 2014, pp. 580–587.
[9] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, "SSD: Single Shot Multibox Detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 21–37.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[13] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

[14] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[15] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125.

[16] F. Zhang, F. Yang, C. Li, and G. Yuan, "CMNet: A Connect-and-Merge convolutional neural network for fast vehicle detection in urban traffic surveillance," *IEEE Access*, vol. 7, pp. 72660–72671, 2019.

[17] K.-J. Kim, P.-K. Kim, Y.-S. Chung, and D.-H. Choi, "Multi-scale detector for accurate vehicle detection in traffic surveillance data," *IEEE Access*, vol. 7, pp. 78311–78319, 2019.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, pp. 1904–1916, May 2015.

[19] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proc. 17th Int. Conf. Pattern Recognit.*, vol. 2. 2004, pp. 28–31.

[20] Z. Zivkovic and F. van der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognit. Lett.*, vol. 27, no. 7, pp. 773–780, May 2006.

[21] S. Suzuki and K. Abe, "Topological structural analysis of digitized binary images by border following," *Comput. Vis., Graph., Image Process.*, vol. 29, no. 3, p. 396, Mar. 1985.

[22] M. Vargas, J. M. Milla, S. L. Toral, and F. Barrero, "An enhanced background estimation algorithm for vehicle detection in urban traffic scenes," *IEEE Trans. Veh. Technol.*, vol. 59, no. 8, pp. 3694–3709, Oct. 2010.

[23] G. Yan, M. Yu, Y. Yu, and L. Fan, "Real-time vehicle detection using histograms of oriented gradients and AdaBoost classification," *Optik*, vol. 127, no. 19, pp. 7941–7951, Oct. 2016.

[24] J. Zhou, D. Gao, and D. Zhang, "Moving vehicle detection for automatic traffic monitoring," *IEEE Trans. Veh. Technol.*, vol. 56, no. 1, pp. 51–59, Jan. 2007.

[25] S. Kul, S. Eken, and A. Sayar, "Distributed and collaborative real-time vehicle detection and classification over the video streams," *Int. J. Adv. Robot. Syst.*, vol. 14, no. 4, Jul. 2017, Art. no. 172988141772078.

[26] Y. Wang, X. Ban, H. Wang, D. Wu, H. Wang, S. Yang, S. Liu, and J. Lai, "Detection and classification of moving vehicle from video using multiple spatio-temporal features," *IEEE Access*, vol. 7, pp. 80287–80299, 2019.

[27] J. Wei, J. Zhao, Y. Zhao, and Z. Zhao, "Unsupervised anomaly detection for traffic surveillance based on background modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 129–136.

[28] X. Hu, X. Xu, Y. Xiao, H. Chen, S. He, J. Qin, and P.-A. Heng, "SINet: A scale-insensitive convolutional neural network for fast vehicle detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 3, pp. 1010–1019, Mar. 2019.

[29] F. Hong, C.-H. Lu, C. Liu, R.-R. Liu, and J. Wei, "A traffic surveillance multi-scale vehicle detection object method base on encoder-decoder," *IEEE Access*, vol. 8, pp. 47664–47674, 2020.

[30] S. Kul, S. Eken, A. Sayar, and Y. Becerikli, "Performance evaluation of support vector machine and convolutional neural network algorithms in real-time vehicle type and color classification," *Evol. Intell.*, vol. 13, no. 1, pp. 83–91, Mar. 2020.

[31] Y. Zhou, L. Liu, L. Shao, and M. Mellor, "Fast automatic vehicle annotation for urban traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 6, pp. 1973–1984, Jun. 2018.

[32] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst. Man Cybern.*, vol. 9, no. 1, pp. 62–66, Jan. 2007.

[33] A. Prati, I. Mikic, M. M. Trivedi, and R. Cucchiara, "Detecting moving shadows: Algorithms and evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 7, pp. 918–923, Jul. 2003.

[34] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[35] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*. [Online]. Available: http://arxiv.org/abs/1602.07360

[36] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[37] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," 2017, *arXiv:1707.01083*. [Online]. Available: http://arxiv.org/abs/1707.01083

[38] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1800–1807.

[39] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2015, pp. 1–14.

[42] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015, *arXiv:1512.00567*. [Online]. Available: http://arxiv.org/abs/1512.00567

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*. [Online]. Available: http://arxiv.org/abs/1512.03385

[44] Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, "CDnet 2014: An expanded change detection benchmark dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2014, pp. 387–394.

[45] A. Sanin, C. Sanderson, and B. C. Lovell, "Shadow detection: A survey and comparative evaluation of recent methods," *Pattern Recognit.*, vol. 45, no. 4, pp. 1684–1695, Apr. 2012.

[46] L. Wen, "UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking," in *Proc. Comput. Vis. Image Understand.*, 2015, pp. 1–4.

[47] A. B. Godbehere, A. Matsukawa, and K. Goldberg, "Visual tracking of human visitors under variable-lighting conditions for a responsive audio art installation," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2012, pp. 4305–4312.

[48] L. Guo, D. Xu, and Z. Qiang, "Background subtraction using local SVD binary pattern," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2016, pp. 86–94.

[49] P. Kaew, T. Pong, and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Video-Based Surveillance Systems*. Boston, MA, USA: Springer, 2002, pp. 135–144. [Online]. Available: https://link.springer.com/book/10.1007%2F978-1-4615-0913-4

ZHIYUAN WANG was born in Langfang, Hebei, China, in 1996. He received the B.S. degree from Guangzhou University, in 2018. He is currently pursuing the M.S. degree with the College of information, Mechanical and Electrical Engineering, Shanghai Normal University, Shanghai. His current research interests include object detection, computer vision, and deep learning.

JIFENG HUANG received the B.S. degree in radio engineering from Zhengzhou University, Zhengzhou, Henan, China, in 1984, the M.S. degree in communication and electronic system from Xi'an Jiaotong University, Xi'an, Shaanxi, China, in 1989, and the Ph.D. degree in measurement and control technology and automation instrument from the East China University of Science and Technology, Shanghai, China, in 2006.

From 1989 to 1999, he was a Teacher with the Zhengzhou University of Aeronautics, Zhengzhou. Since 1999, he has been a Professor with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University. His research interests include pattern recognition, machine learning, and automation instrument.

Dr. Huang received the Award for Scientific and Technological Advancement from the Aviation Ministry of China.

NEAL N. XIONG (Senior Member, IEEE) received the Ph.D. degree in sensor system engineering from Wuhan University, in 2007, and the Ph.D. degree in dependable communication networks from the Japan Advanced Institute of Science and Technology, in 2008.

He worked with Georgia State University, Wentworth Technology Institution, and Colorado Technical University (Full Professor for a period of five years) for a period of ten years. He is currently an Associate Professor with the Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK, USA. He has published over 200 international journal articles and over 100 international conference papers. Some of his works were published in the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE or ACM TRANSACTIONS, ACM Sigcomm workshop, the IEEE INFOCOM, ICDCS, and IPDPS. His research interests include cloud computing, security and dependability, parallel and distributed computing, networks, and optimization theory.

Dr. Xiong has been a Senior Member of the IEEE Computer Society, since 2012. He is the Chair of the Trusted Cloud Computing Task Force, the IEEE Computational Intelligence Society (CIS), and the Industry System Applications Technical Committee. He has received the Best Paper Award in the 10th IEEE International Conference on High Performance Computing and Communications (HPCC-08) and the Best Student Paper Award in the 28th North American Fuzzy Information Processing Society Annual Conference (NAFIPS2009). He has been a General Chair, a Program Chair, a Publicity Chair, a Program Committee Member, and an Organizing Committee Member of over 100 international conferences. He has been a Reviewer of about 100 international journals, including the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS (PART: A/B/C), the IEEE TRANSACTIONS ON COMMUNICATIONS, the IEEE TRANSACTIONS ON MOBILE COMPUTING, and the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS. He is serving as an Editor-in-Chief, an Associate Editor or an Editor Member for over ten international journals (including Associate Editor for the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, an Associate Editor for *Information Science*, the Editor-in-Chief for the *Journal of Internet Technology* (JIT), and the *Journal of Parallel & Cloud Computing* (PCC)), and a Guest Editor for over ten international journals, including *Sensor Journal*, *WINET*, and *MONET*.

XIAOPING ZHOU was born in Shanghai, China, in 1978. He received the B.S. degree in electrical engineering from the Nanchang University of China, Nanchang, in 2001, the M.S. degree in electrical engineering from the Chongqing University of Posts and Telecommunications, Chongqing, in 2006, and the Ph.D. degree from Shanghai University, Shanghai, in 2011. From 2011 to 2013, he was a Postdoctoral Fellow with the Communication Laboratory, Shanghai Jiao Tong University of Science and Technology, China. He is currently an Associate Professor with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University. His current research interests include deep learning, pattern recognition, and computer vision.

XIAO LIN (Member, IEEE) is currently a Professor with the College of Information, Mechanical and Electrical Engineering, Shanghai Normal University. Her current research interests include machine learning, computer vision, and image processing.

THEODORE LEE WARD is currently an Assistant Professor with the Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, OK, USA. His research interests include cloud computing, wireless LAN, and time-domain analysis.