IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 15, 2022, accepted January 27, 2022, date of publication January 31, 2022, date of current version February 8, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3147866

Intelligent Traffic-Monitoring System Based on YOLO and Convolutional Fuzzy Neural Networks

CHENG-JIAN LIN[1,2], (Senior Member, IEEE), AND JYUN-YU JHANG[2,3]

[1]Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung 41170, Taiwan
[2]College of Intelligence, National Taichung University of Science and Technology, Taichung 40401, Taiwan
[3]Department of Computer Science and Information Engineering, National Taichung University of Science and Technology, Taichung 40401, Taiwan

Corresponding author: Cheng-Jian Lin (cjlin@ncut.edu.tw)

This work was supported by the Ministry of Science and Technology of the Republic of China, Taiwan, under Contract MOST 110-2221-E-167 -031 -MY2.

ABSTRACT With the rapid pace of urbanization, the number of vehicles traveling between cities has increased significantly. Consequently, many traffic-related problems have emerged, such as traffic jams and excessive numbers and types of vehicles. To solve traffic problems, road data collection is important. Therefore, in this paper, we develop an intelligent traffic-monitoring system based on you only look once (YOLO) and a convolutional fuzzy neural network (CFNN), which record traffic volume, and vehicle type information from the road. In this system, YOLO is first used to detect vehicles and is combined with a vehicle-counting method to calculate traffic flow. Then, two effective models (CFNN and Vector-CFNN) and a network mapping fusion method are proposed for vehicle classification. In our experiments, the proposed method achieved an accuracy of 90.45% on the Beijing Institute of Technology public dataset. On the GRAM-RTM data set, the mean average precision and F-measure (F1) of the proposed YOLO-CFNN and YOLO-VCFNN vehicle classification methods are 99%, superior to those of other methods. On actual roads in Taiwan, the proposed YOLO-CFNN and YOLO-VCFNN methods not only have a high F1 score for vehicle classification but also have outstanding accuracy in vehicle counting. In addition, the proposed system can maintain a detection speed of more than 30 frames per second in the AGX embedded platform. Therefore, the proposed intelligent traffic monitoring system is suitable for real-time vehicle classification and counting in the actual environment.

INDEX TERMS Traffic-monitoring system, fuzzy neural network, vehicle classification, feature fusion, deep learning.

I. INTRODUCTION

Road traffic monitoring is an important research topic. By analyzing the types of vehicles and traffic flow on the road, current traffic conditions can be understood, and actionable information can be provided to traffic management agencies. This information can help these agencies to make decisions that improve people's quality of life. For example, on holidays, information regarding the road traffic volume can be used to suggest alternate routes to drivers to divert traffic from congested areas. In addition, if large trucks often use a certain road, roadside warnings can be installed to alert drivers and reduce traffic accidents. Moreover, the type and color of a specific vehicle can be used to identify and track the vehicles of criminals. The abovementioned applications all rely on information collected by a road monitoring system for analysis. Therefore, to obtain information on passing vehicles, many researchers have used different methods to achieve vehicle detection and classification.

Traditional vehicle-detection methods are mainly divided into two types: (1) Static-based methods [1]–[7] that use sliding windows or shape feature comparison methods to generate vehicle prediction frames and verify them based on the information in the prediction frames and (2) methods that use the dynamic features of a moving object [8]–[12] to separate it from the image to obtain the contour of the object. Regarding static-based methods, Mohamed *et al.* [1] proposed a vehicle-detection system that uses Haar-like features to extract vehicle shape features and inputs the extracted features into an artificial neural network to realize vehicle classification. Wen *et al.* [2] also used Haar-like features to extract the edge and structural features of vehicles and input them

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

into AdaBoost to filter important features. Then, the filtered features were input into a support vector machine (SVM) for classification to improve its recognition accuracy. Sun *et al.* [3] and David and Athira [4] used Garbor filters to obtain vehicle characteristics and then input them into an SVM to determine whether a vehicle is present in an image. Wei *et al.* [5] designed a two-step vehicle-detection method. First, they used Haar-like features and AdaBoost to obtain the region of interest with vehicles and subsequently used the histogram of oriented gradients (HOG) [6] and an SVM to reverify the region. According to their experimental results, their method exhibited improved vehicle-detection capability. Yan *et al.* [7] designed a vehicle-detection system that used vehicle shadows to select the boundaries of vehicles and the HOG to extract features. These features were then input into an AdaBoost classifier and SVM classifier for verification. In this method, when vehicles block each other, they are regarded as one vehicle because the shadows are connected to each other, which weakens the detection effect.

In terms of dynamics, Seenouvong *et al.* [8] proposed a vehicle-detection and counting system based on dynamic features. Background subtraction was used to obtain a difference map from a given current image to achieve segmentation of the corresponding foreground image. In addition, various morphological operations were used to obtain the outline and bounding box of a moving object, detect moving vehicles, and count the vehicles passing through a designated area. A few researchers have used Gaussian mixture models (GMMs) [9], [10] to model the background or adaptive background [11]–[13] with the aim of solving the problem of background subtraction due to background images. Poor foreground segmentation is caused by gradual changes in brightness. The aforementioned static and dynamic methods have many limitations in overcoming this problem. For example, traditional feature extraction methods must be manually designed by experts on the basis of their experience, meaning that the process is complicated. Moreover, the extracted features are mostly pieces of shallow vertical and horizontal information, which cannot effectively describe the changes in vehicle features and cannot be widely used. The dynamic feature method increases the complexity of subsequent image processing operations in cases of extensive background changes in addition to yielding poor detection results. With recent advancements in deep learning, these conventional methods have gradually been replaced by deep learning techniques.

II. LITERATURE REVIEW

In recent years, deep learning has been widely used in many fields, and good prediction results have been obtained with this method. Compared with traditional methods that require artificial feature determination, the convolutional neural network (CNN) method greatly improves the accuracy of image recognition. Initially, Lecun *et al.* [14] proposed the LeNet model to solve the problem of recognizing handwritten digits in the banking industry. Krizhevsky *et al.* [15] proposed AlexNet to improve the traditional CNN by deepening the model architecture and using the ReLU excitation function and the dropout layer to increase the effectiveness of the network during learning and prevent overfitting. Szegedy *et al.* [16] proposed GoogLeNet, which uses multiple filters of different sizes to extract features that enrich feature information. Simonyan and Zisserman [17] proposed two models, namely VGG-16 and VGG-19. They replaced the large convolution kernel by successively using multiple small convolution kernels to perform operations and proved that increasing the depth of a model can improve its accuracy. He *et al.* [18] proposed the ResNet model. They used residual blocks to solve the problem of gradient disappearance and convergence inability due to excessive network depth. Howard *et al.* [19] proposed MoblieNet, which uses deep separation convolution to extract fewer and more useful features and reduces the number of redundant parameters in a CNN model.

The aforementioned studies have focused on improving the feature description capabilities of a CNN to extend the application of CNNs to more complex problems, such as object detection. Several researchers [20]–[24] have used region-based CNN (R-CNN) series models to solve the vehicle-detection problem. R-CNN uses the region proposal network (RPN) [25] to extract the position of an object and then classifies it by using a traditional CNN. RetinaNet [26] is the latest network architecture of R-CNN models. The R-CNN framework comprises a two-stage mechanism and uses a multilayer neural network for classification [27], [28]. This architecture substantially increases the number of parameters used and decreases the execution speed; thus, it is unsuitable for real-time detection. To solve this problem, one-stage mechanism methods have been proposed for vehicle detection, such as the you-only-look-once (YOLO) framework model [29]–[31] and the single-shot multibox detector (SSD) [32] framework model. One-stage methods are fast and can detect objects in real time, but their classification accuracy is lower than that of R-CNN methods [33], [34].

The aforementioned object-detection methods have the following problems: 1) Two-stage object-detection methods have high classification accuracy, but the large of network parameters decrease the detection speed. 2) One-stage object-detection methods have a high real-time detection speed but lower accuracy than two-stage object-detection methods. 3) To increase the number of object categories, the entire network must be retrained, which is time-consuming and reduces the scalability of the method.

Recently, fuzzy neural networks (FNNs) [35]–[39] that have a human-like fuzzy inference mechanism and the powerful learning functions of neural networks have been widely used in various fields, such as classification, control, and forecasting. Asim *et al.* [35] applied an adaptive network-based fuzzy inference system to classification problems. Compared with traditional neural networks, this method yielded higher classification accuracy. Lin *et al.* [36] used an interval type-2 FNN and tool chips to predict flank wear, and their method

yielded superior prediction results. A few researchers have used a locally recurrent functional link fuzzy neural network [37] and Takagi–Sugeno–Kang-type FNNs [38], [39] to solve system identification and prediction problems, and both methods have yielded good results. In this study, an FNN was embedded into a deep learning network to reduce the number of parameters used in the network and obtain superior classification results. Conventional CNNs use pooling, global pooling [40], and channel pooling [41] methods for feature fusion. Global pooling methods sum the spatial information and perform operations on each feature map to achieve feature fusion and can be divided into global average pooling (GAP) [42] and global max pooling (GMP) [43]. Thus, global pooling methods are more robust to spatial translations of the input and prevent overfitting. Channel pooling methods include channel average pooling (CAP) [44] and channel max pooling (CMP) [45], which perform feature fusion by computing average or maximum pixel values, respectively, at the same positions in each channel of feature maps. Furthermore, these methods only compress features and do not contain learnable weights, leading to poor classification results. In this study, a new feature fusion method named network mapping was proposed to enhance the utility of feature fusion and explore the effectiveness of different feature fusion methods.

To design an intelligent traffic-monitoring system with fast execution speed, high classification accuracy, and high category extensibility, a two-stage object-detection method was adopted in this study. The proposed intelligent traffic-monitoring system based on YOLO and a convolutional FNN (CFNN) collects real-time information on traffic volume and vehicle type on the road. In this system, a novel modified YOLOv4-tiny (mYOLOv4-tiny) is first used to detect vehicles and is then combined with a vehicle counting method to calculate the traffic flow. Furthermore, two effective models (CFNN and Vector-CFNN) and a network mapping fusion method that improve the computational efficiency, classification accuracy, and category extensibility were proposed for vehicle classification. The proposed model architecture has fewer network parameters compared to other models; therefore, the system can achieve real-time, high-accuracy vehicle classification with limited hardware resources and flexible extensibility for different categories.

The contributions of this study can be summarized as follows:

- An intelligent traffic-monitoring system was developed to record real-time information about traffic volume, and vehicle types.
- An mYOLOv4-tiny model was proposed to achieve real-time object detection and improve detection efficiency.
- Two effective models (CFNN and Vector-CFNN) that adopt a new network mapping fusion method were implemented to increase the classification accuracy and greatly reduce the number of model parameters.
- Category extensions (e.g., vehicle type) only require training of the classification model (CFNN) without retraining of the object detection model (YOLO). This not only saves substantial training time but also improves the flexibility of category extension.
- The proposed intelligent traffic monitoring system was implemented on the NVIDIA AGX Xavier embedded platform and applied to provincial highway 1 (T362) in Kaohsiung, Taiwan for real-time vehicle tracking, counting, and classification.

The remainder of this paper is organized as follows: Section 3 introduces the proposed YOLO-CFNN method for intelligent traffic monitoring. The experimental results of the proposed method are described in Section 4. Section 5 presents our conclusions and an outline of future work.

III. PROPOSED YOLO-CFNN FOR INTELLIGENT TRAFFIC MONITORING

In this section, an intelligent traffic-monitoring system is introduced. The proposed system has three functions, namely (1) vehicle detection, (2) vehicle counting, and (3) vehicle classification. The system architecture is illustrated in Fig. 1.

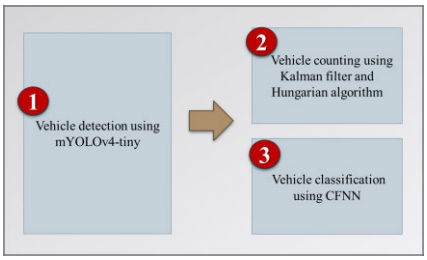


FIGURE 1. Three functions of the proposed intelligent traffic-monitoring system.

A flowchart of the proposed intelligent traffic-monitoring system is presented in Fig. 2. First, real-time road images are obtained from traffic cameras. Then, the proposed mYOLOv4-tiny model is used to detect the position of a vehicle. To solve the problem of the repeated recording of the same car as different vehicles in different frames,

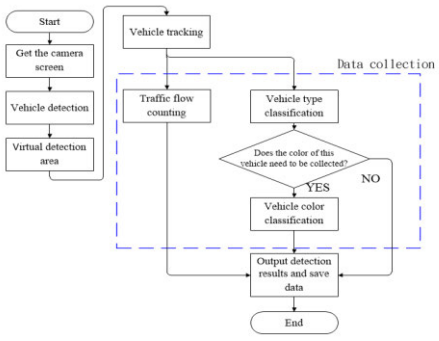


FIGURE 2. Flowchart of the proposed intelligent traffic-monitoring system.

FIGURE 3. Virtual detection area.

a counting algorithm is introduced to track the vehicle. In other words, a vehicle is assigned the same identity (ID) across different frames. Before execution of the counting algorithm, the virtual detection area (as shown in Fig. 3) of the target vehicle is screened to reduce the computational burden. Finally, the vehicles passing through the virtual detection area are counted and classified, and the resulting information is collected and stored for subsequent analysis.

A. VEHICLE DETECTION USING MODIFIED YOLOv4-tiny

The conventional YOLOv4-tiny is a lightweight network simplified using YOLO. It uses convolutional layers and max pooling layers to extract object features. In addition, YOLOv4-tiny uses UpSampling and Concat layers to merge features and expand feature information to further improve detection. Compared with other YOLO and SSD methods, YOLOv4-tiny has a faster detection speed. However, the detection accuracy of YOLOV4-tiny is worse than that of YOLO and SSD methods due to its greatly simplified network architecture. Conventional YOLOv4-tiny uses two outputs for object detection. To improve the detection accuracy, an mYOLOv4-tiny that has three outputs was designed for vehicle detection. The network architecture of mYOLOv4-tiny is depicted in Fig. 4. In total, 24 convolutional layers and three max pooling layers were used. Finally, three scales—25×15, 50×30, and 100×60—were used for prediction. In this system, the mYOLOv4-tiny model is only used to detect vehicle objects.

B. VEHICLE COUNTING METHOD

The above-described YOLO object detection method can be used to identify a vehicle and its location information from a single picture. However, in actual traffic applications, a continuous image frame is provided as the input. The vehicles detected in different image frames are independent of each other. Therefore, the same vehicle is counted multiple times, and the collected vehicle information would be wrong. To solve this problem, the ID of the detected vehicle must be configured to prevent double counting. In the proposed system, an object counting method is added to correlate and match the vehicles detected in different image frames and to determine whether a detected vehicle is newly added. In this study, the multiobject counting method [46] is adopted, which uses vehicle position information from the previous frame obtained by the detection method to predict the position of a vehicle in the current frame by applying a Kalman filter. Then, the actual vehicle position detected in the current frame and the current vehicle position estimated using the Kalman filter are used to calculate the intersection over union (IoU) as the distance cost. Finally, the Hungarian algorithm is applied to match vehicles to achieve vehicle tracking.

C. VEHICLE CLASSIFICATION USING CFNN

By using the methods described in Subsections 2.1 and 2.2, the vehicle position can be determined from the complete image and the vehicle can be segmented. Next, to collect more detailed information, such as vehicle type, the segmented vehicle image is analyzed and the results are obtained after classification. If information items must be added, the YOLO model need not be retrained; thus, it has superior extensibility and reduced training time after category expansion.

To identify relevant vehicle information, two CFNNs, called CFNN and Vector-CFNN, are proposed, as illustrated

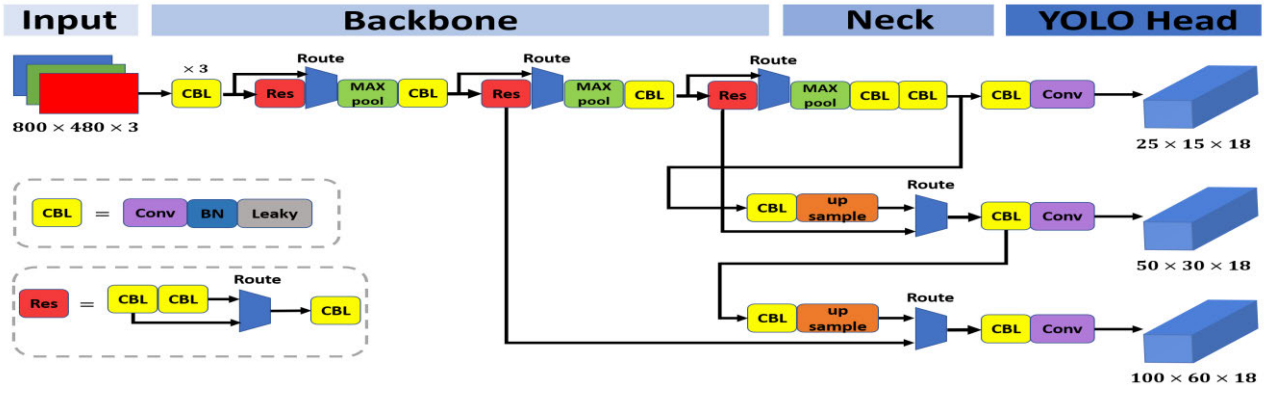


FIGURE 4. Network architecture of mYOLOv4-tiny.

in Fig. 5. In the CFNN model in Fig. 5(a), at the outset, the convolutional layer is used to extract features from the image, and the maximum pooling layer is then used to compress these features to reduce the amount of calculation. The interactive stacking method is used to increase the model depth to complete various shape feature combinations, and a feature fusion layer is added to reduce the dimensionality of the feature size and integrate information. Finally, the fused feature information is sent to the FNN for classification to obtain the classification result of vehicle type. To solve the problem of multiple redundant parameters in the traditional CNN model, this study proposes a Vector-CFNN model (i.e., Fig. 5(b)). The architecture of this model is similar to that of CFNN, and the traditional convolutional layer is replaced with a two-layer vector kernel convolutional layer [47] to further reduce the number of parameters and computational complexity of the model.

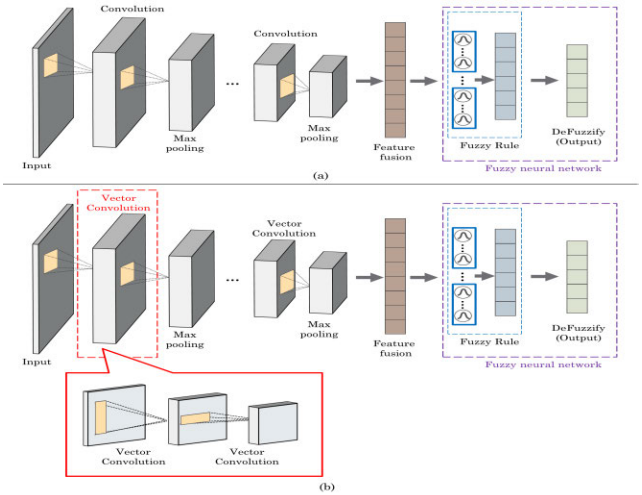


FIGURE 5. Schematic of the proposed network architecture: (a) CFNN model and (b) Vector-CFNN model.

Next, the feature fusion layer and FNN classifier are explained in the proposed models.

1) FEATURE FUSION LAYER

In the feature fusion layer, different fusion methods can be used to integrate different types of feature information to obtain more useful features. Given a large number of input features, a suitable fusion method is selected to compress the features and reduce the dimensionality of the information between them. For method selection, the features are fused using either pooling operations or network mapping. Based on the different operation rules between features, different fusion results can be obtained, as summarized in Table 1.

In this study, a network mapping fusion method is proposed. This method assigns a weight to the information of each extracted feature and then integrates these weights to obtain new features. The calculation method is shown in Fig. 6, and the calculation formula is as follows:

$$f_z = \sum_{i=1}^{n} w_{zi} * x_i. \quad (1)$$

where f_z is the output of the zth fusion, n is the total number of input features, x_i is the ith input feature element, and w_{zi} is the ith input weight used in the zth fusion result.

TABLE 1. Different fusion methods.

Fusion methods	Operation	Description
Global pooling fusion	Maximum operation Average operation	Calculate each feature and fuse the entire feature map (width and height) separately.
Channel pooling fusion	Maximum operation Average operation	Fusion of all channels (depth) between feature maps.
Network mapping fusion	Weighted product operation	Each element in the feature map uses different weights for fusion.

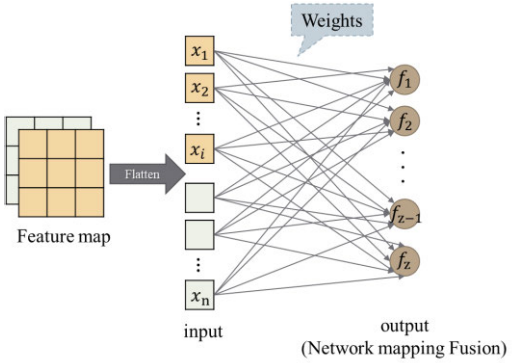


FIGURE 6. Schematic of the network mapping fusion method.

2) FNNS

FNNs mimic human logical thinking and learning abilities. In terms of network design, an FNN can be divided into the input layer, fuzzification layer, rule layer, and defuzzification layer. The fuzzy set is contained in the fuzzy layer, and its members can have different degrees of membership on the interval is [0, 1]; this is known as a membership function. The fuzzy membership function converts input data to a value in [0, 1] based on the degree of membership of a specified set, providing a measure of the degree of similarity of an element of a fuzzy set. Common fuzzy membership functions include triangular, trapezoidal, bell-shaped, and Gaussian; among these, the Gaussian membership function has the highest accuracy [48]. Therefore, the Gaussian function is adopted as the membership function in the proposed CFNN. The feature vectors extracted by convolution operation are classified by a FNN. The If–then can be used to represent the fuzzy rules to make fuzzy inferences (Fig. 7).

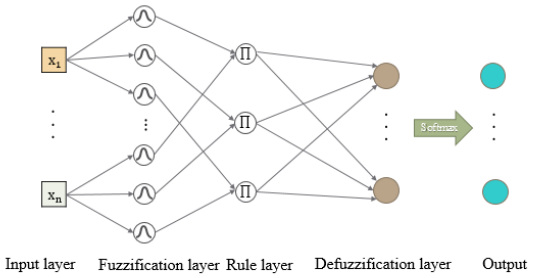


FIGURE 7. Schematic of a fuzzy neural network.

IV. EXPERIMENTAL RESULTS

To verify the effectiveness of the proposed intelligent traffic-monitoring system, three experiments were performed in this study. Section 4A describes the evaluation indicators and the parameter settings of the CFNN model. Section 4B compares the classification efficiency of various CNN models and feature fusion methods by using the Beijing Institute of Technology Vehicle data set (BIT-Vehicle Dataset). Section 4C applies the public GRAM road-traffic monitoring (GRAM-RTM) data set to compare the evaluation indicators between the proposed YOLO-CFNN and the state-of-the-art object detection methods. Section 4D presents the proposed intelligent traffic monitoring system that was implemented in the AGX Xavier embedded platform and applied to provincial highway 1 (T362) in Kaohsiung, Taiwan for vehicle classification, tracking, and counting.

A. EXPERIMENTAL DESIGN

To evaluate the output results of the model, this study used the category with the highest model output value (top-1) as the classification result and accuracy as the evaluation indicator. The calculation formula is as follows:

$$accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{2}$$

where TP, FP, TN, and FN denote true positive, false positive, true negative, and false negative, respectively. The mean average precision (mAP), precision, recall, F-measure (F1), and detection speed (FPS) were also adopted to verify the effectiveness of various object detection models. The evaluation indicators can be calculated as follows:

$$mAP = \frac{\sum_{k=1}^{k=n} AP_k}{n} \tag{3}$$

$$precision = \frac{TP}{TP + FP} \tag{4}$$

$$recall = \frac{TP}{TP + FN} \tag{5}$$

$$F1 = \frac{2precision \times recall}{precision + recall} \tag{6}$$

$$FPS = \frac{frame}{second} \tag{7}$$

Here, n indicates the number of classes, and AP_k denotes the (AP) of class k. In the experimental environment, TensorFlow and Keras were used as the deep learning environment and developmental tool, respectively, and an RTX2080Ti graphics card was used to train the network model. The parameter settings of the proposed CFNN and Vector-CFNN models are summarized in Tables 2 and 3, respectively.

TABLE 2. Parameter settings of the proposed CFNN model.

Layer	Kernel Size	Number of filters	Stride
Input	-	-	-
Convolution_1	3x3	32	2,2
Max_pooling_1	2x2	-	2,2
Convolution_2	3x3	64	1,1
Max_pooling_2	2x2	-	2,2
Convolution_3	3x3	128	1,1
Max_pooling_3	2x2	-	2,2
Convolution_4	3x3	64	1,1
Max_pooling_4	2x2	-	2,2
Feature fusion	-	128	-
Fuzzy Rule Layer	-	64	-
DeFuzzify Layer	-	Number of categories	-

TABLE 3. Parameter settings of the proposed Vector-CFNN model.

Layer	Kernel Size	Number of filters	Stride
Input	-	-	-
Convolution_1-1	3x1	32	2,1
Convolution_1-2	1x3	32	1,2
Max_pooling_1	2x2	-	2,2
Convolution_2-1	3x3	64	1,1
Convolution_2-2	1x3	64	1,1
Max_pooling_2	2x2	-	2,2
Convolution_3-1	3x1	128	1,1
Convolution_3-2	1x3	128	1,1
Max_pooling_3	2x2	-	2,2
Convolution_4-1	3x1	64	1,1
Convolution_4-2	1x3	64	1,1
Max_pooling_4	2x2	-	2,2
Feature fusion	-	128	-
Fuzzy Rule Layer	-	64	-
DeFuzzify Layer	-	Number of categories	-

In the CFNN model, the input image size is set to 224 × 224 × 3, and four sets of convolutional layers and pooling layers are used to achieve feature extraction. In each convolutional layer uses a 3 × 3 (see Table 1), 3 × 1, or 1 × 3 (see Table 2) convolution kernel to extract features. Each feature is compressed through the largest pooling layer of size 2 × 2 to reduce the computational load. In the convolutional layer, 32, 64, and 128 are used as the number of convolution kernels in the first three layers to extract various shape feature combinations. Then, the number of convolution kernels in the last layer is set to 64, and the feature fusion layer

TABLE 4. Number of each vehicle type.

Vehicle types	Bus	Minibus	Minivan	Sedan	SUV	Truck
Quantity	558	883	476	5,921	1,392	823
Training	200	200	200	200	200	200
Testing	200	200	200	200	200	200

FIGURE 8. Vehicle images after segmentation in BIT dataset.

is added to reduce the dimensionality of the features. Here, by using the proposed network mapping method, a total of 128 features are fused and input into the FNN for classification. The output size represents the number of categories of different vehicles.

B. CLASSIFICATION RESULTS OF BIT-VEHICLE DATASET

The BIT-Vehicle Dataset [49] is a public dataset for vehicle classification collated by Beijing Institute of Technology. The dataset contains a total of 9,850 vehicle images, all of which were captured using 2 cameras at different times and locations on highways. These images differ in brightness, proportions, and surface color. The dataset includes six vehicle types: buses, minibuses, minivans, sedans, sports utility vehicles (SUVs), and trucks. Each image contains 1 or 2 vehicles. The vehicle positions marked in advance in the dataset are segmented (Fig. 8), and the vehicle types and numbers after segmentation are listed in Table 4.

In the training and testing of the model, according to the processing method described in [49], 200 vehicles were randomly selected from each category to be the training and testing data. In total, 2400 images each were used as the training and test datasets for the experiment. Ten experiments were performed using these data sets and the average of the values obtained in these experiments was used for evaluation. This study used different fusion methods to evaluate the performance of the proposed CFNN and Vector-CFNN models. The experimental results are listed in Table 5. The accuracies of the CFNN and Vector-CFNN models reached 90.20% and 90.45%, respectively, with the network mapping fusion method. Compared with the global pooling and channel pooling methods, the proposed network mapping fusion method has higher accuracy.

Moreover, the two proposed models were compared with other common models, namely AlexNet, GoogLeNet, VGG-16, VGG-19, ResNet50, Sparse Laplacian CNN [49], and PCN-Net [50]. The experimental comparison results are summarized in Table 6. According to the table, the accuracy of the

TABLE 5. Experimental results of CFNN and Vector-CFNN models with various feature fusion methods.

Feature fusion methods			Worst accuracy	Best accuracy	Average accuracy
CFNN	Global Pooling	Average	85.50%	87.83%	86.61%
		Max	80.67%	89.00%	85.23%
	Channel Pooling	Average	81.75%	85.42%	83.69%
		Max	81.75%	85.17%	83.68%
	Network Mapping		85.67%	92.00%	90.20%
Vector-CFNN	Global Pooling	Average	78.75%	86.17%	82.91%
		Max	79.08%	87.25%	84.42%
	Channel Pooling	Average	81.50%	85.42%	83.02%
		Max	82.92%	85.58%	84.53%
	Network Mapping		89.50%	91.25%	90.45%

TABLE 6. Experimental results of vehicle classification using various models.

Models		Average accuracy	The amount of parameters (10^6)
AlexNet [15]		85.33%	29.99
GoogLeNet [16]		87.08%	3.23
VGG-16 [17]		86.25%	65.08
VGG-19 [17]		86.58%	70.38
ResNet50 [18]		87.08%	21.98
Sparse Laplacian CNN [49]		88.11%	1.38
PCN-Net [50]		88.52%	0.653
Proposed method	CFNN	90.20%	0.315
	Vector-CFNN	90.45%	0.28

TABLE 7. Number of vehicles in the training and testing phases using GRAM-RTM.

Vehicle types	Big-truck	Truck	Car	Van	Motorcycle
Quantity	914	805	1063	896	754
Training	712	694	814	670	600
Testing	202	111	249	226	154

two CFNN models is higher than that of other deep learning classification methods. The accuracy of the two CFNN and Vector-CFNN models is 0.89% and 1.93% higher, respectively, than that of PCN-Net, and 51.7% and 57.1% fewer parameters are used in the CFNN and Vector-CFNN models, respectively, than in PCN-Net.

C. VEHICLE CLASSIFICATION RESULTS ON THE GRAM-RTM DATA SET

The GRAM-RTM (M-30) data set [51] was used to compare the performance of the proposed YOLO-CFNN and state-of-the-art object detection methods, including RetinaNet, SSD, YOLOv4, and YOLOv4 tiny. The M30 contains 7520 frames with a resolution of 800 × 480 at 30 fps recorded using a

TABLE 8. Vehicle classification results for the proposed YOLO-CFNN with various object detection methods.

Method	Backbone	Big-truck	Truck	Car	Van	Motorcycle	Precision	Recall	F1	mAP	FPS
Retinanet	ResNet50	98.99%	97.53%	94.44%	93.14%	92.21%	98.45%	54.48%	35.1%	95.26%	16
SSD	VGG16	99.61%	98.85%	98.79%	98.93%	97.55%	97.54%	91.49%	94.11%	98.15%	25
YOLOv4	CSPNet	100%	99.79%	99.81%	99.83%	99.54%	97.46%	97.75%	97.6%	99.79%	44
YOLOv4 tiny	CSPNet	99.71%	99.8%	89.39%	96.93%	66.01%	96.975	85.1%	90.64%	90.37%	291
YOLO-CFNN	CSPNet	100%	100%	98.8%	100%	98.44%	99.71	99.51%	99.6%	99.45%	73
YOLO-VCFNN	CSPNet	100%	100%	98.77%	100%	98.44%	99.4%	99.49%	99.44%	99.44%	78

TABLE 9. Specifications of the NVIDIA Jetson AGX Xavier platform.

Specification	Content
GPU	512-core NVIDIA Volta GPU with Tensor Cores
CPU	8-core NVIDIA Carmel ARM 64-bit CPU
Memory	16GB LPDDR4x 2133MHz
Storage	32GB eMMC 5.1
Size	87 mm x 100 mm

Nikon Coolpix L20 camera. Vehicle types include large truck, truck, car, van, and motorcycle. The ratio of training data to test data was 8:2. That is, 80% of the data (6016 frames) were used for training and 20% of the data (1504 frames) were used for testing. The number of vehicles in the training and testing phases are presented in Table 7. First, the vehicle object detection model (mYOLOv4-tiny) was trained, followed by the classification models (CFNN and Vector-CFNN). The vehicle classification results are listed in Table 8, which reveals that the proposed YOLO-CFNN, YOLO-VCFNN, and YOLOv4 yield a mAP as high as 99%. The proposed YOLO-CFNN and YOLO-VCFNN methods had a higher F1 score than other models.

The traditional YOLOv4-tiny model had a higher detection speed (300 FPS) but lower F1 and mAP than other models. However, the proposed two-stage vehicle classification models (YOLO-CFNN and YOLO-VCFNN) achieved a high score for the evaluation indicators and a detection speed of over 70 FPS. Thus, the proposed YOLO-CFNN and YOLO-VCFNN models can be employed for real-time vehicle classification applications.

D. APPLICATION TO PROVINCIAL HIGHWAY 1 (T362) IN KAOHSIUNG

To verify the effectiveness of the system in an actual environment, the proposed intelligent traffic-monitoring system was applied to roads in Taiwan, and its architecture is depicted in Fig. 9. In this architecture, each monitored road section houses a camera and an AGX Xavier embedded computing platform. The specifications of the AGX Xavier platform are listed in Table 9. Real-time images are processed using the AGX Xavier platform to obtain detailed traffic data, and the data are sent to the road monitoring center for analysis

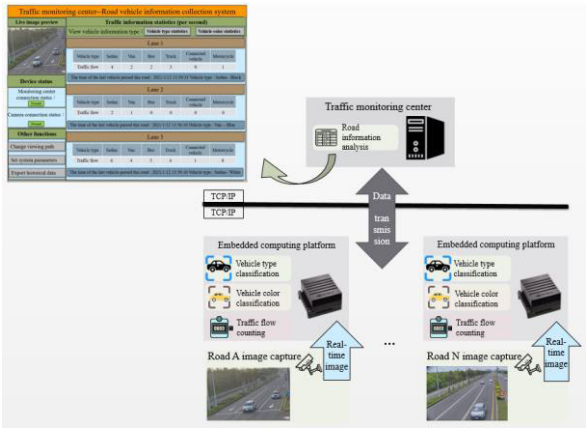


FIGURE 9. Intelligent traffic-monitoring system in Taiwan.

through the wireless network. Therefore, three functions are achieved on the road monitored for testing, namely vehicle classification, and traffic flow calculation. Moreover, a T362 vehicle data set, which contained vehicle type, is established for Kaohsiung, Taiwan to verify the performance of the proposed model.

The number of vehicle types image data points in the T362 vehicle dataset was 1,815. The images were captured from different lanes and at different times. Therefore, the captured images were illuminated by different light sources, and some vehicles were blocked, as illustrated in Fig. 10. The T362 vehicle dataset has the following six vehicle types: buses, trucks, cars, motorcycles, and trailers. The numbers of each vehicle type are listed in Table 10. In terms of model training and testing, this study used 80% of the collected vehicle type dataset as the training data and 20% as the testing data. The input image size of the classification model was uniformly adjusted to 224 × 224 × 3, and 10 experimental runs were performed to ensure the stability of the experiment.

1) CLASSIFICATION RESULTS FOR VEHICLE TYPE

In the evaluation conducted using the vehicle type dataset, the experimental results obtained using CNN and Vector-CNN with different fusion methods are summarized in Table 11. The CFNN and Vector-CFNN models with the proposed network mapping fusion method exhibited the best accuracy values of 94.68% and 95.28%, respectively. The proposed

FIGURE 10. Samples of collected vehicle images in the T362 vehicle dataset.

TABLE 10. Numbers of each vehicle type.

Vehicle type	Bus	Truck	Car	Motorcycle	Trailer
Quantity	109	435	572	139	88
Training	87	348	457	111	70
Testing	22	87	115	28	18

TABLE 11. Vehicle type classification results obtained using CFNN and Vector-CFNN with different fusion methods.

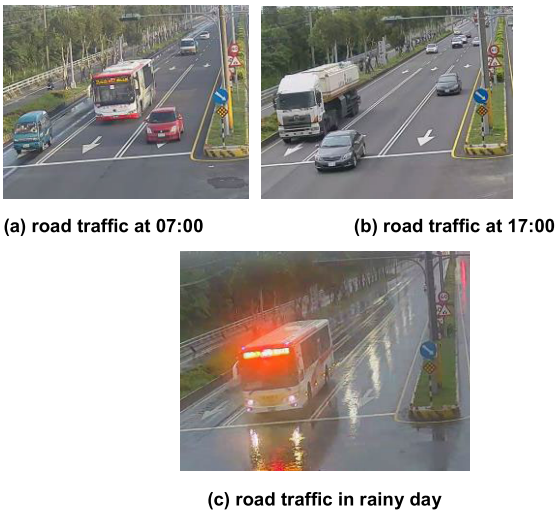
Feature fusion methods			Worst accuracy	Best accuracy	Average accuracy
CFNN	Global Pooling	Average	85.50%	87.83%	86.61%
		Max	80.67%	89.00%	85.23%
	Channel Pooling	Average	81.75%	85.42%	83.69%
		Max	81.75%	85.17%	83.68%
	Network Mapping		85.67%	92.00%	90.20%
Vector-CFNN	Global Pooling	Average	78.75%	86.17%	82.91%
		Max	79.08%	87.25%	84.42%
	Channel Pooling	Average	81.50%	85.42%	83.02%
		Max	82.92%	85.58%	84.53%
	Network Mapping		89.50%	91.25%	90.45%

TABLE 12. Vehicle type classification results obtained using various models.

Models		Worst accuracy	Best accuracy	Average accuracy	The amount of parameters (10^6)
AlexNet		92.60%	94.79%	93.56%	46.7
VGG-16		90.41%	93.15%	91.97%	134.2
MobileNet		77.53%	92.32%	85.61%	4.2
LeNet		83.01%	89.86%	87.73%	5.2
Proposed method	CFNN	93.15%	95.89%	94.68%	0.5
	Vector-CFNN	94.24%	96.16%	95.28%	0.5

network mapping method is superior to the other fusion methods in vehicle classification.

The CFNN and Vector-CFNN models proposed in this study were compared with a few common deep learning methods, and the experimental results are listed in Table 12. The accuracy of the proposed models with the network mapping fusion method was superior to that of the other classification methods. The accuracy of the proposed models was 1.83%, 3.59%, 8.6%, and 11.29% higher than the accuracy of AlexNet, VGG16, LeNet, and MoblieNet, respectively. In terms of the number of parameters, the proposed CFNN and Vector-CFNN models had the lowest parameter value of approximately 0.5 M. In addition, compared with the lightweight MobileNet, LeNet, and AlexNet, the two proposed CFNN models reduced the number of parameters by up to 86.8%, 89.4%, and 98.8%, respectively. Thus, the proposed models achieve favorable classification and offer a competitive advantage when few parameters are used.



(a) road traffic at 07:00 (b) road traffic at 17:00

(c) road traffic in rainy day

FIGURE 11. Three actual road traffic videos.

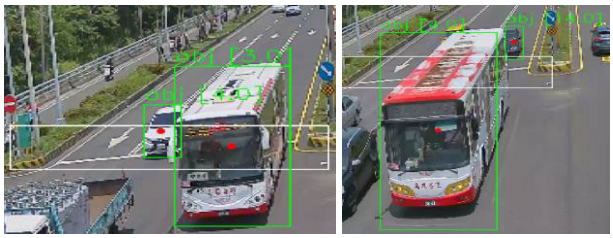


FIGURE 12. Different occlusion conditions in a real road scene.

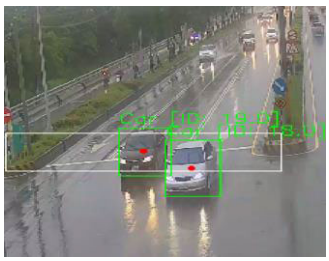


FIGURE 13. The visual vehicle detection and counting.

2) COUNTING RESULTS OF ACTUAL ROAD TRAFFIC FLOW

Finally, the vehicle counting method used in this study was evaluated and verified. In this experiment, the same

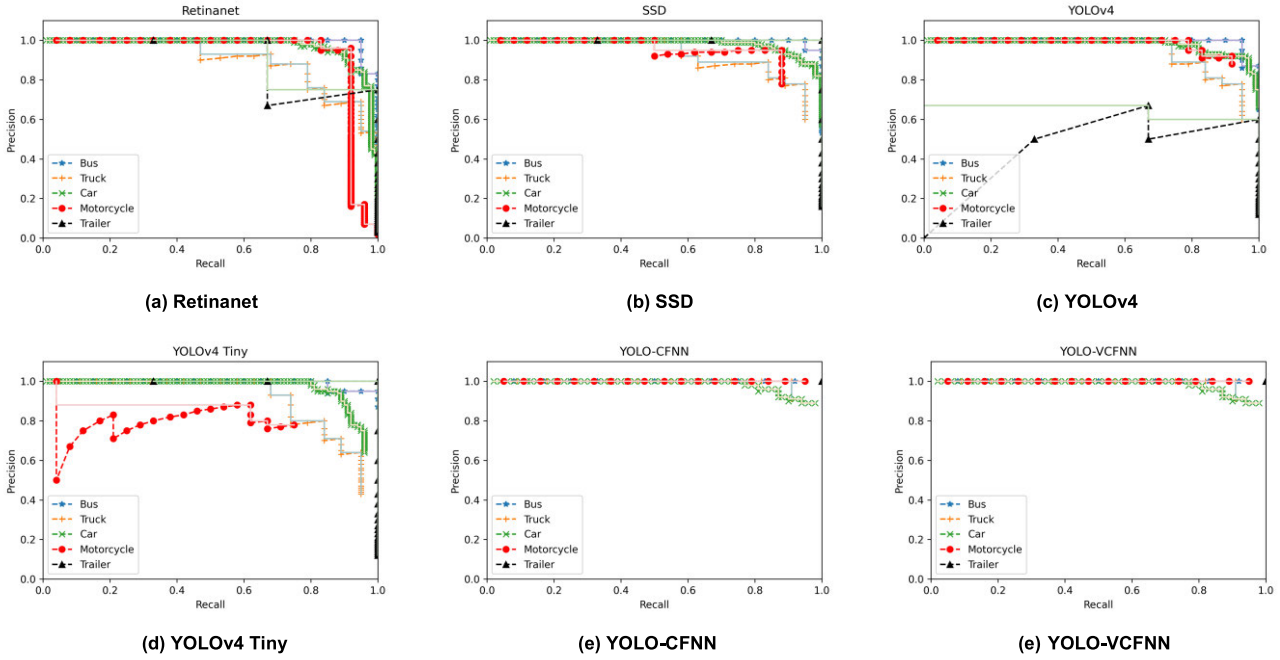


FIGURE 14. Precision vs. recall curves of the various detection methods by using actual road traffic videos at 7:00.

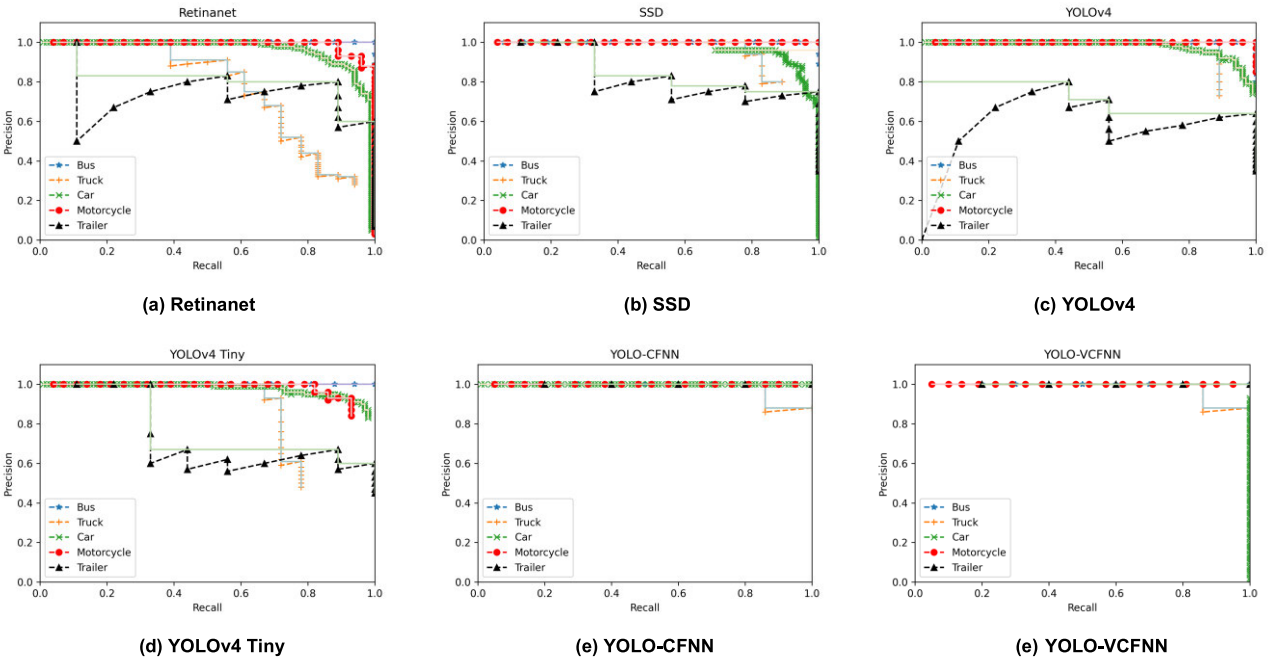


FIGURE 15. Precision versus recall curves of the various detection methods by using actual road traffic videos at 17:00.

traffic scene was used for verification. Three actual road traffic videos were used to evaluate the proposed vehicle counting method. Each video was 5 min long, and the two selected videos were recorded at 07:00 and 17:00. The remaining videos were taken in rainy conditions. Still images from the three videos are displayed in Fig. 11.

In the evaluation, the proposed vehicle flow counting result was divided by the manual counting result to determine the accuracy of vehicle counting. In addition, different occlusion

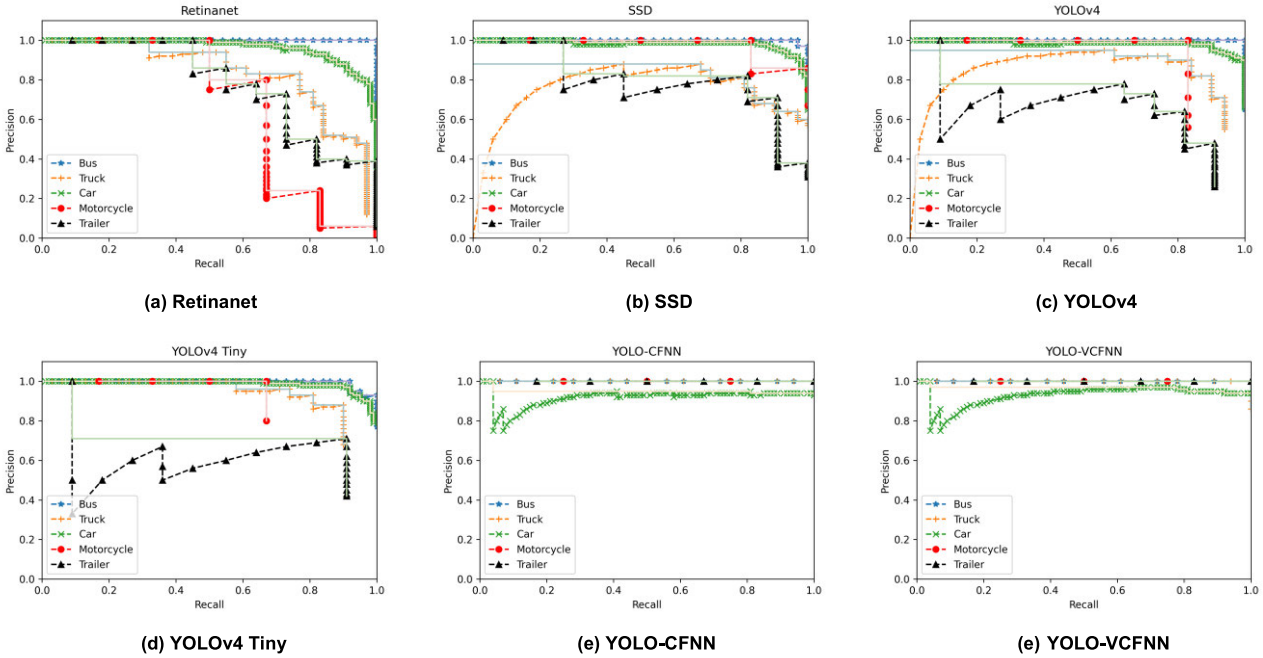


FIGURE 16. Precision versus recall curves of the various detection methods using actual road traffic videos for rainy conditions.

TABLE 13. Traffic flow counting results obtained using actual road traffic videos at 7:00.

Method	Bus	Truck	Car	Motorcycle	Trailer	Precision	Recall	F1	mAP	FPS	Accuracy of flow counting
Retinanet	99.17%	90.36%	97.04%	92.31%	91.67%	84.83%	68.94%	76.06%	94.11%	8	—
SSD	99.76%	89.85%	98.29%	85.8%	100%	90.39%	82.54%	86.28%	94.74%	12	—
YOLOv4	99.35%	91.42%	98.13%	90.77%	64.44%	84.29%	86.86%	85.55%	88.82%	22	—
YOLOv4 tiny	99.29%	88.87%	94.29%	65.49%	100%	85.33%	80.93%	83.07%	89.59%	145	—
YOLO-CFNN	91.67%	90.91%	96.68%	94.74%	100%	96.33%	94.31%	99.6%	95.3%	33	97.05% (33/34)
YOLO-VCFNN	91.67%	96.1%	89.09%	94.18%	100%	91.13%	95.35%	99.44%	93.19%	36	97.05% (33/34)

conditions were included in the real road scene as presented in Fig. 12. As shown in Fig. 12, a larger bus blocks a car, resulting in a missed count. The visual vehicle detection and counting are shown in Fig. 13. The text in the first half of the green label in Fig. 13 represents the type of vehicle and the text in the second half represents the number of counts. When a vehicle enters the virtual detection zone, the proposed intelligent traffic-monitoring system immediately performs vehicle classification and counting.

The traffic flow counting results of each video are summarized in Tables 13–15. The precision versus recall curves of the proposed YOLO-CFNN and YOLO-VCFNN models are shown in Figs. 14–16. As shown in Table 13, the mAP of RetinaNet and SSD was 94%, but the F1 scores were only 76.06% and 86.28%, respectively. The mAP and F1 score of YOLOv4 were 88.82% and 85.55%, respectively. However, the mAP for trailers was only 64.44%. Although YOLOv4-tiny has a detection speed of 145 FPS, the motorcycle detection performance was poor (65.49%). The proposed YOLO-CFNN and YOLO-VCFNN are superior to other methods in terms of F1 score (99%). After introducing the counting method into CFNN and VCFNN, FPS can be maintained above 30 to achieve real-time detection. The two proposed methods also had an accuracy of 97.05% in traffic flow vehicle counting.

For the afternoon road traffic video (Table 14), the mAP and F1 of YOLO-CFNN and YOLO-VCFNN were higher than those of other methods. The accuracy of flow counting was 98.5%. For the rain video (Table 15), except for the SSD method, the mAP of the motorcycle detection was lower because images captured in rainy conditions are blurry,

TABLE 14. Traffic flow counting results obtained using actual road traffic videos at 17:00.

Method	Bus	Truck	Car	Motorcycle	Trailer	Precision	Recall	F1	mAP	FPS	Accuracy of flow counting
Retinanet	100%	75.67%	96.22%	99.06%	81.48%	86.97%	71.15%	76.06%	78.26%	7	—
SSD	100%	87.43%	94.8%	100%	85.8%	91.75%	83.39%	86.28%	87.37%	13	—
YOLOv4	100%	88.89%	96.85%	100%	72.06%	89.53%	89.66%	85.55%	89.59%	21	—
YOLOv4 tiny	100%	75.21%	96.18%	92.2%	77.04%	90.16%	84.7%	83.07%	87.34%	148	—
YOLO-CFNN	90%	98.21%	100%	95.24%	100%	97.5%	97.04%	99.6%	97.26%	32	98.5% (66/67)
YOLO-VCFNN	100%	98.21%	92.77%	95.24%	100%	90.77%	97.6%	99.44%	94.06%	35	98.5% (66/67)

TABLE 15. Traffic flow counting results obtained using actual road traffic videos for rainy conditions.

Method	Bus	Truck	Car	Motorcycle	Trailer	Precision	Recall	F1	mAP	FPS	Accuracy of flow counting
Retinanet	100%	83.57%	96.28%	68.34%	78.68%	83.47%	69.47%	75.82%	85.38%	7	—
SSD	99.93%	82.74%	98.55%	97.62%	82.12%	87.91%	86.26%	87.07%	92.19%	10	—
YOLOv4	100%	86.49%	98.7%	68.3%	83.33%	85.76%	83.73%	84.73%	87.36%	18	—
YOLOv4 tiny	99.45%	87.96%	97.66%	66.67%	67.53%	88.87%	81.17%	84.84%	83.85%	140	—
YOLO-CFNN	94.44%	100%	94.73%	75%	100%	97.63%	93.88%	95.71%	92.83%	30	100% (47/47)
YOLO-VCFNN	88.89%	100%	96.7%	75%	100%	97.83%	91.66%	94.64%	92.12%	33	100% (47/47)

affecting the judgment results. However, the mAP and F1 of the two proposed methods were higher than 90%, and the counting accuracy was 100%. These scenarios reveal that the proposed intelligent traffic-monitoring system is suitable for real-time vehicle counting in actual environments and has a high counting accuracy.

V. CONCLUSION

In this study, an intelligent traffic-monitoring system was proposed to calculate traffic flows and classify vehicle types. The major contributions of this study are as follows:

- A novel intelligent traffic-monitoring system combining a YOLOv4-tiny model and counting method was proposed for traffic volume statistics and vehicle type classification.
- The proposed CFNN and Vector-CFNN were designed by introducing the fusion method and FNN, which can not only effectively reduce the number of network parameters, but also enhance the classification accuracy.
- The proposed network mapping fusion method was superior to the commonly used pooling method, and it could effectively integrate image features and improve the classification accuracy.
- Compared with the current state-of-the-art object detection methods (Retinanet, SSD, YOLOv4, and YOLOv4 tiny), the proposed YOLO-CFNN and YOLO-VCFNN have a high mAP rate, accurate counting accuracy, and real-time vehicle counting and classification ability (over 30FPS).

The experimental results indicated that the performance of the proposed CFNN and Vector-CFNN models was superior to that of common deep learning models. On the BIT dataset, compared with the pooling method, the proposed network mapping fusion method improved the recognition accuracy by 3.59%–5.92%. In addition, compared with the PCN-Net model, the proposed CFNN and Vector-CFNN models improved the accuracy by 1.93% and reduced the number of parameters by 57.1%. On the GRAM-RTM data set, the mAP and F1 of the two proposed vehicle classification methods were 99%, higher than those of other methods. In addition, among the FPS indicators, the proposed method was 1.65 times faster than the traditional YOLOv4. On the T362 vehicle type dataset, compared with the general pooling methods, the accuracy of the proposed network mapping fusion method was 2.3%–5.36% higher. In addition, compared with the AlexNet model, the accuracy of the proposed CFNN and Vector-CFNN models was 1.19% and

1.83% higher, respectively, and the number of parameters decreased by 98.8%. In three actual road traffic scenarios, the proposed YOLO-CFNN and YOLO-VCFNN methods yielded a high F1 score for vehicle classification and high accuracy for vehicle counting. In summary, the CFNN and Vector-CFNN models proposed in this study not only have favorable vehicle classification effects but also have fewer parameters relative to other models. Therefore, the proposed models are suitable for information analysis in environments with limited hardware performance.

In terms of the extensibility of the proposed models, many factors that affect the machining accuracy of machine tools in intelligent manufacturing have been identified, such as temperature and tool wear. Therefore, developing an accurate model of the effects of these factors is crucial. In future studies, the proposed CFNN and Vector-CFNN models and the network mapping fusion method will be applied for modeling in intelligent manufacturing.

REFERENCES

[1] A. Mohamed, A. Issam, B. Mohamed, and B. Abdellatif, "Real-time detection of vehicles using the Haar-like features and artificial neuron networks," *Proc. Comput. Sci.*, vol. 73, pp. 24–31, Jan. 2015.

[2] X. Wen, L. Shao, W. Fang, and Y. Xue, "Efficient feature selection and classification for vehicle detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 3, pp. 508–517, Mar. 2015.

[3] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection using Gabor filters and support vector machines," in *Proc. 14th Int. Conf. Digit. Signal Process. (DSP)*, Jul. 2002, pp. 1019–1022.

[4] H. David and T. A. Athira, "Improving the performance of vehicle detection and verification by log Gabor filter optimization," in *Proc. 4th Int. Conf. Adv. Comput. Commun.*, Aug. 2014, pp. 50–55.

[5] Y. Wei, Q. Tian, J. Guo, W. Huang, and J. Cao, "Multi-vehicle detection algorithm through combining Harr and HOG features," *Math. Comput. Simul.*, vol. 155, pp. 130–145, Jan. 2018.

[6] S. Bougharriou, F. Hamdaoui, and A. Mtibaa, "Linear SVM classifier based HOG car detection," in *Proc. 18th Int. Conf. Sci. Techn. Autom. Control Comput. Eng. (STA)*, Dec. 2017, pp. 241–245.

[7] 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, 2016.

[8] N. Seenouvong, U. Watchareeruetai, C. Nuthong, K. Khongsomboon, and N. Ohnishi, "A computer vision based vehicle detection and counting system," in *Proc. 8th Int. Conf. Knowl. Smart Technol. (KST)*, Feb. 2016, pp. 224–227.

[9] P. K. Bhaskar and S.-P. Yong, "Image processing based vehicle detection and tracking method," in *Proc. Int. Conf. Comput. Inf. Sci. (ICCOINS)*, Jun. 2014, pp. 1–5.

[10] N. Seenouvong, U. Watchareeruetai, C. Nuthong, K. Khongsomboon, and N. Ohnishi, "Vehicle detection and classification system based on virtual detection zone," in *Proc. 13th Int. Joint Conf. Comput. Sci. Softw. Eng. (JCSSE)*, Jul. 2016, pp. 1–5.

[11] M. Anandhalli and V. P. Baligar, "Improvised approach using background subtraction for vehicle detection," in *Proc. IEEE Int. Advance Comput. Conf. (IACC)*, Jun. 2015, pp. 303–308.

[12] N. S. Sakpal and M. Sabnis, "Adaptive background subtraction in images," in *Proc. Int. Conf. Adv. Commun. Comput. Technol. (ICACCT)*, Feb. 2018, pp. 439–444.

[13] N. Shah, A. Pingale, V. Patel, and N. V. George, "An adaptive background subtraction scheme for video surveillance systems," in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT)*, Dec. 2017, pp. 13–17.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[15] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst. (NIPS)*, vol. 1, Dec. 2012, pp. 1097–1105.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.

[18] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[19] 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*.

[20] K. Shi, H. Bao, and N. Ma, "Forward vehicle detection based on incremental learning and fast R-CNN," in *Proc. 13th Int. Conf. Comput. Intell. Secur. (CIS)*, Dec. 2017, pp. 73–76.

[21] S.-C. Hsu, C.-L. Huang, and C.-H. Chuang, "Vehicle detection using simplified fast R-CNN," in *Proc. Int. Workshop Adv. Image Technol. (IWAIT)*, Jan. 2018, pp. 1–3.

[22] S. Rujikietgumjorn and N. Watcharapinchai, "Vehicle detection with subclass training using R-CNN for the UA-DETRAC benchmark," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–5.

[23] W. Zhang, Y. Zheng, Q. Gao, and Z. Mi, "Part-aware region proposal for vehicle detection in high occlusion environment," *IEEE Access*, vol. 7, pp. 100383–100393, 2019.

[24] L. Wang, Y. Lu, H. Wang, Y. Zheng, H. Ye, and X. Xue, "Evolving boxes for fast vehicle detection," in *Proc. IEEE Int. Conf. Multimedia Expo. (ICME)*, Jul. 2017, pp. 1135–1140.

[25] 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.

[26] 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.

[27] N. A. Al-Sammarraie, Y. M. H. Al-Mayali, and Y. A. Baker El-Ebiary, "Classification and diagnosis using back propagation artificial neural networks (ANN)," in *Proc. Int. Conf. Smart Comput. Electron. Enterprise (ICSCEE)*, Jul. 2018, pp. 1–5.

[28] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, Nov. 2018, Art. no. e00938.

[29] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.

[30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[31] Z. Jiang, L. Zhao, S. Li, and Y. Jia, "Real-time object detection method based on improved YOLOv4-tiny," 2020, *arXiv:2011.04244*.

[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, Oct. 2016, pp. 21–37.

[33] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/Accuracy trade-offs for modern convolutional object detectors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3296–3297.

[34] P. Soviany and R. T. Ionescu, "Optimizing the trade-off between single-stage and two-stage deep object detectors using image difficulty prediction," in *Proc. 20th Int. Symp. Symbolic Numeric Algorithms Sci. Comput. (SYNASC)*, Sep. 2018, pp. 209–214.

[35] Y. Asim, B. Raza, A. K. Malik, A. R. Shahid, M. Faheem, and Y. J. Kumar, "A hybrid adaptive neuro-fuzzy inference system (ANFIS) approach for professional bloggers classification," in *Proc. 22nd Int. Multitopic Conf. (INMIC)*, Nov. 2019, pp. 1–6.

[36] C.-J. Lin, J.-Y. Jhang, S.-H. Chen, and K.-Y. Young, "Using an interval type-2 fuzzy neural network and tool chips for flank wear prediction," *IEEE Access*, vol. 8, pp. 122626–122640, 2020.

[37] D. K. Bebarta, R. Bisoi, and P. K. Dash, "Locally recurrent functional link fuzzy neural network and unscented H-infinity filter for shortterm prediction of load time series in energy markets," in *Proc. IEEE Power, Commun. Inf. Technol. Conf. (PCITC)*, Oct. 2015, pp. 663–670.

[38] J.-W. Yeh and S.-F. Su, "Efficient approach for RLS type learning in TSK neural fuzzy systems," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2343–2352, Sep. 2017.

[39] C.-J. Lin, C.-H. Lin, and J.-Y. Jhang, "Dynamic system identification and prediction using a self-evolving Takagi–Sugeno–Kang-type fuzzy CMAC network," *Electronics*, vol. 9, no. 4, p. 631, Apr. 2020.
[40] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*.
[41] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z. Si, and J. Guo, "Fine-grained vehicle classification with channel max pooling modified CNNs," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3224–3233, Apr. 2019.
[42] V. Christlein, L. Spranger, M. Seuret, A. Nicolaou, P. Kral, and A. Maier, "Deep generalized max pooling," in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1090–1096.
[43] Z. Li, S.-H. Wang, R.-R. Fan, G. Cao, Y.-D. Zhang, and T. Guo, "Teeth category classification via seven-layer deep convolutional neural network with max pooling and global average pooling," *Int. J. Imag. Syst. Technol.*, vol. 29, no. 4, pp. 577–583, May 2019.
[44] Z. Gao, Y. Li, Y. Yang, N. Dong, X. Yang, and C. Grebogi, "A coincidence-filtering-based approach for CNNs in EEG-based recognition," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 7159–7167, Nov. 2020.
[45] L. Cheng, D. Chang, J. Xie, R. Ma, C. Wu, and Z. Ma, "Channel max pooling for image classification," in *Intelligence Science and Big Data Engineering. Visual Data Engineering*, Z. Cui, J. Pan, S. Zhang, L. Xiao, and J. Yang, Eds. Cham, Switzerland: Cham, Switzerland: Springer, 2019, pp. 273–284.
[46] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, p. 346.
[47] J. Ou and Y. Li, "Vector-kernel convolutional neural networks," *Neurocomputing*, vol. 330, pp. 253–258, Feb. 2019.
[48] N. Talpur, M. N. M. Salleh, and K. Hussain, "An investigation of membership functions on performance of ANFIS for solving classification problems," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 226, Aug. 2017, Art. no. 012103.
[49] Z. Dong, Y. Wu, M. Pei, and Y. Jia, "Vehicle type classification using a semisupervised convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2247–2256, Aug. 2015.
[50] F. C. Soon, H. Y. Khaw, J. H. Chuah, and J. Kanesan, "Semisupervised PCA convolutional network for vehicle type classification," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8267–8277, Aug. 2020.
[51] R. Guerrero-Gómez-Olmedo, R. J. López-Sastre, S. Maldonado-Bascón, and A. Fernández-Caballero, "Vehicle tracking by simultaneous detection and viewpoint estimation," in *Natural and Artificial Computation in Engineering and Medical Applications*, J. M. F. Vicente, J. R. Sánchez, F. de la Paz López, F. J. T. Moreo, Eds. Berlin, Germany: Springer, 2013, pp. 306–316.

CHENG-JIAN LIN (Senior Member, IEEE) received the B.S. degree in electrical engineering from the Tatung Institute of Technology, Taipei, Taiwan, in 1986, and the M.S. and Ph.D. degrees in electrical and control engineering from the National Chiao Tung University, Taiwan, in 1991 and 1996, respectively. Currently, he is a Chair Professor with the Computer Science and Information Engineering Department, National Chin-Yi University of Technology, Taichung, Taiwan, and the Dean of the Intelligence College, National Taichung University of Science and Technology, Taichung. His current research interests include machine learning, pattern recognition, intelligent control, image processing, intelligent manufacturing, and evolutionary robot.

JYUN-YU JHANG received the B.S. and M.S. degrees from the Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung, Taiwan, in 2015, and the Ph.D. degree in electrical and control engineering from the National Yang Ming Chiao Tung University, Taiwan, in 2021. He is an currently an Assistant Professor with the Computer Science and Information Engineering Department, National Taichung University of Science and Technology, Taichung. His current research interests include fuzzy logic theory, type-2 neural fuzzy systems, evolutionary computation, machine learning, and computer vision and application.