Received December 1, 2019, accepted December 19, 2019, date of publication January 1, 2020, date of current version January 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2963486

Vehicle Type Classification Using an Enhanced Sparse-Filtered Convolutional Neural Network With Layer-Skipping Strategy

SURYANTI AWANG[1], NIK MOHAMAD AIZUDDIN NIK AZMI[1], AND MD. ARAFATUR RAHMAN[1,2,3], (Senior Member, IEEE)

[1]Soft Computing & Intelligent Systems (SPINT), Faculty of Computing, Universiti Malaysia Pahang, Kuantan 26300, Malaysia
[2]IBM Center of Excellence, Universiti Malaysia Pahang, Kuantan 26300, Malaysia
[3]Earth Resources and Sustainability Center, Universiti Malaysia Pahang, Kuantan 26300, Malaysia

Corresponding author: Suryanti Awang (suryanti@ump.edu.my)

This work was supported in part by Universiti Malaysia Pahang, Malaysia, under Grant RDU190315.

ABSTRACT In this paper, a vehicle type classification approach is proposed by using an enhanced feature extraction technique based on Sparse-Filtered Convolutional Neural Network with Layer-Skipping strategy (SF-CNNLS). To extract rich and discriminant vehicle features, we introduce Three-Channels of SF-CNNLS (TC-SF-CNNLS) as the feature extraction technique. Local and global features of vehicles are extracted from three channels of an image which are, luminance and chromatic components. This technique is inspired by how human eyes differentiating objects that share almost similar features. TC-SF-CNNLS is tested with a benchmark dataset that provides frontal-view images to classify vehicle types of the bus, passenger car, taxi, minivan, SUV, and truck with Softmax Regression as a classifier. This test aims to observe the ability of this technique in differentiating vehicles with almost similar features but different classes. A test is also conducted with the self-obtained dataset (SPINT) to observe the effectiveness of this technique. The results are observed based on accuracy, precision, recall, and f-score, whereby, TCSF-NNLS has successfully recognized all the classes with an average accuracy of 0.905, precision is between 0.8629 to 0.9548, recall is between 0.83 to 0.96 and f-score is between 0.8564 to 0.9523. In addition, this technique is able to outperform other existing techniques with an average accuracy of 93.% compared to only 89.2% when 5 classes of vehicles are tested.

INDEX TERMS Vehicle type recognition, convolutions neural network, deep learning, computational intelligence.

I. INTRODUCTION

Vehicle type classification is one of the applications that is able to increase the efficiency of road and transportation infrastructure. This application can be implemented in various related systems, for instance, Automatic Toll Collection (ATC), Vehicle Counting System (VCS) and Traffic Monitoring System [1], [2]. The systems can increase the efficiency of many related things including traffic census, traffic surveillance, traffic control, and forecast. The application can be grouped into camera-based or sensor-based. This paper will focus on the camera-based whereby a vehicle is classified based on a processed vehicle image.

Nowadays, traffic surveillance cameras are provided everywhere in big or medium cities to assist in the monitoring process. The process can be more efficient with the implementation of Artificial Intelligence applications, for instance, vehicle type classification. However, a vehicle from the same class can appear in various appearances, and a vehicle from a different class may have a similar appearance. For instance, passenger car and taxi, as well as bus and truck share almost similar features, especially frontal-view images. These variations make visual recognition becomes challenging.

In a vehicle type classification, classifying each vehicle type according to the class that has been determined by the road and transportation ministry is crucial in order to provide an accurate result for further implementation. In Malaysia, the determined vehicle classes are passenger car, taxi, SUV, van, minivan, lorry, truck, and bus. However, due to the challenges that have been mentioned, most of the related studies provide a classification of general classes. For example,

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

passenger cars and taxis are classified as sedan class, and trucks and buses are classified as heavy vehicles.

The challenges are difficult to be solved by extracting local and global features of a vehicle from a single image component which is a grayscale image as implemented in existing CNN techniques. This is because the extracted features are limited and the unique features that differentiate the intra-class vehicles are unable to be extracted. Due to that, this paper proposes to enhance the feature extraction technique by implementing the three channels of an image into the unsupervised and supervised CNN known as TC-SF-CNNLS. This technique will extract local and global features from the channels of an image. The channels are based on luminance and two chromatic components. The three channels implementation is inspired by how human eyes are able to differentiate an object based on the shape, color, and brightness [3]. This proposed technique aims to extract richer and unique features of vehicles with the hypothesis that the unique features are able to differentiate the vehicle type based on the respective determined classes. Thus, it will overcome the challenge mentioned above and produce an accurate classification performance.

In the related works section, the related studies will be discussed to show the trend of vehicle type classification research, followed by the overview of proposed technique that explains about the implementation of TC-SF-CNNLS technique. Section IV is about the experiments and results, and the last section is the conclusion.

II. RELATED WORKS

Vehicle type classification system based on surveillance camera offers many benefits to the society, for instance, enhancing the security, and road enforcement efficiency. In order to do the classification, vehicle features can be extracted from a grayscale image or color image from the camera. It is believed that richer information of the vehicle features can be gathered using a color image compared to the grayscale image. Moreover, the vehicle types are divided into various classes. Reported studies classify the vehicle type into various classes. The classes are sedan car, minivan, truck, bus, microbus, SUV, light vehicle, and heavy vehicle.

Numerous studies have revealed that there are various techniques that have been implemented in order to develop vehicle type classification systems. The most prevalent techniques are based on the deep learning method that has a significant advantage in extracting low-level and high-level features from an image to improve object classification and recognition performance [4]. Convolutional Neural Networks (CNN) is the most popular technique that has been implemented in vehicle type classification due to its advantages [5]–[8]. CNN can be implemented based on supervised, unsupervised, semi-supervised, or a combination of them.

For example, Dong *et al.* [9], [10] extracted vehicle features from grayscale images by introducing unsupervised and semi-supervised CNN, respectively. They achieved 83% to 98% of accuracies based on vehicle classes of the sedan, minivan, microbus, truck, and bus from their own dataset, named BIT dataset. However, based on the provided results, they classified all types of car and taxi as a sedan and did not include certain cars such as cars with sunroof in their experiment. This is similar to Bautista *et al.* [11] that achieved 94.72% of accuracy performance when they used the same technique to classify vehicle types into classes of jeep, sedan car, bus, SUV and van based on grayscale images from the low quality of surveillance camera. Rong and Xia [12] utilized CNN, that is similar to Dong *et al.* [10] to extract vehicle features from grayscale images. They classified vehicle types into classes of car and van using a vehicle dataset from Caltech that based on surveillance camera [13]. They tested the accuracy performance using Softmax regression and compared with SVM and DBN. Based on the results, Softmax regression achieved the accuracy of 84% that surpassed other classifiers.

A multitask of region-based CNN (R-CNN) technique was implemented by Huo et al. to classify vehicle types into classes of sedan car, van, bus, and truck based on grayscale images [14]. The multitasks in this study were defined by the labels provided in the output layers. The aim was to classify the vehicle into car, truck, bus, and van class with a different angle (front, side, and rear). With this technique, they achieved an accuracy performance of 83% which is quite low. In addition, the false classification rate is quite high in each class with 5% to 10%.

Another implementation of vehicle type classification based on grayscale images from a surveillance camera is done by Wang et al. that combined CNN with the extreme learning technique (ELM) [15]. The vehicle features from grayscale images were extracted using CNN, and additional samples of a vehicle are trained in ELM to extract other vehicle features. An adaptive clustering was used to classify the extracted features into general classes of vehicle. The classes are compact car, mid-size car (van) and heavy-duty. In this study, they considered all types of cars such as SUV as a heavy-duty vehicle, and taxi as in compact car class. 85.56% of the accuracy performance was achieved based on a grayscale image of a vehicle with front and back angles. Huttunen *et al.* [16] implemented CNN on grayscale images. The CNN consists of a convolutional layer and a max pooling layer with a support vector machine (SVM) as a classifier. Based on only four classes of vehicle, namely, bus, truck, van, and small car, the accuracy of over 97% is achieved.

The existing SF-CNNLS technique has been implemented to classify three classes of vehicle type based on grayscale images [17]. SF-CNNLS is an unsupervised and supervised CNN that enhances the filter by using a layer skipping strategy. The vehicles are classified into passenger car, taxi, and truck class. The results are reasonably high with the accuracies are between 85% to 98%. However, the classes are too limited, and the false positive is considered high with between 8% to 11% for taxi class and passenger car class. It is due to the consideration of cars with sunroof in the experiment. Thus, it leads to misclassification as the taxi class.

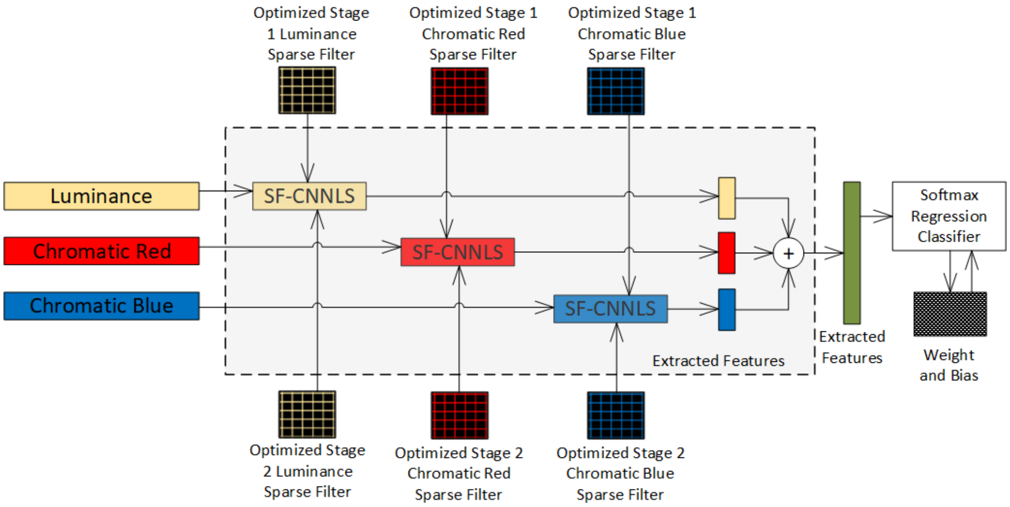

Optimized Stage 1 Luminance Sparse Filter
Optimized Stage 1 Chromatic Red Sparse Filter
Optimized Stage 1 Chromatic Blue Sparse Filter
Luminance
Chromatic Red
Chromatic Blue
SF-CNNLS
SF-CNNLS
SF-CNNLS
+
Extracted Features
Extracted Features
Softmax Regression Classifier
Weight and Bias
Optimized Stage 2 Luminance Sparse Filter
Optimized Stage 2 Chromatic Red Sparse Filter
Optimized Stage 2 Chromatic Blue Sparse Filter

FIGURE 1. The conceptual framework of TC-SF-CNNLS technique.

Besides CNN, there is a study that implemented DBN in the vehicle type classification based on grayscale images [18]. They implemented processes of histogram equalization, edge emphasis, binarization, geometric scaling, and resizing the images and the DBN classifier with four parameters into a grayscale image. The highest accuracy they have achieved is 89.53% when vehicle types of bikes, motorcycles, and other vehicle classes are classified. However, there is no further explanation on what is the vehicle type for the other vehicle classes.

A recent study has implemented feature fusion convolutional neural network (FFCN) to produce discriminative features and to avoid interference caused by environmental factors. However, this study does not exert with recognizing the vehicles based on class types. The study is focused on the recognition of vehicle models, make and year. The similar focus has been implemented by Tian *et al.* [19] using iterative discrimination CNN (ID-CNN).

However, apart from deep learning method, a simple and straight forward approach has been implemented by Lin and Zhao [20]. They implemented k-means clustering to distinguish vehicles into small, medium, and large vehicle class based on distance features of the grayscale vehicle images.

Based on these existing studies, it can be concluded that most of the studies have implemented CNN technique in classifying the vehicle type based on grayscale images into classes of passenger car or sedan car, SUV, van, bus, truck, bikes, and motorcycles. However, none of the related studies has considered color images in classifying the vehicle types into real and specific classes as determined by the authority. Although there is a study to classify the specific classes (car, taxi, and truck), the results are not promising, and the classes are too limited. Hence, there is a need for an optimal solution in the classification and differentiation of the vehicle type using color images to the determined classes.

III. OVERVIEW OF THE PROPOSED TECHNIQUE

An enhanced of SF-CNNLS is proposed by using three channels of an image known as TC-SF-CNNLS technique to classify the vehicle types to overcome the limitation of the existing techniques, as explained in the previous section. Local and global features of a vehicle will be extracted from the luminance and chromatic components of color images of the vehicle. The components are YCrCb that is efficient for a recognition task [21]. Y is the luminance component that has a grayscale color space, and it represents brightness. Cr is the chromatic component known as chromatic red, and Cb is chromatic blue. Based on the characteristics of YCrCb, richer local and global features of the vehicle will be obtained, and they are able to be used in discriminating the vehicle types.

Figure 1 presents the conceptual framework of TC-SF-CNNLS technique. Generally, there are three components of an image. In each component, there is unsupervised and supervised CNN known as SF-CNNLS that is implemented to extract the features. Each component requires a set of optimized sparse filters (sparse filter stage 1 and stage 2). These filters are used to extract the features from the image. The extracted features from the 3 components are concatenated prior to the classification process using Softmax Regression classifier.

The detailed workflow of the proposed technique is depicted in figure 2. The entire system consists of several phases. The main phases are training and testing. The system starts with the image acquisition phase; then, the acquired images are divided into training and testing datasets to be processed in the respective phases. Later, in the training and testing phase, the pre-processing phase is implemented and followed by the feature extraction phase. For the training phase, the feature extraction is done in unsupervised and supervised training with several processes are deployed. Subsequently, the extracted features will be stored in a database. The features will be retrieved to be matched with the extracted features from the testing phase during

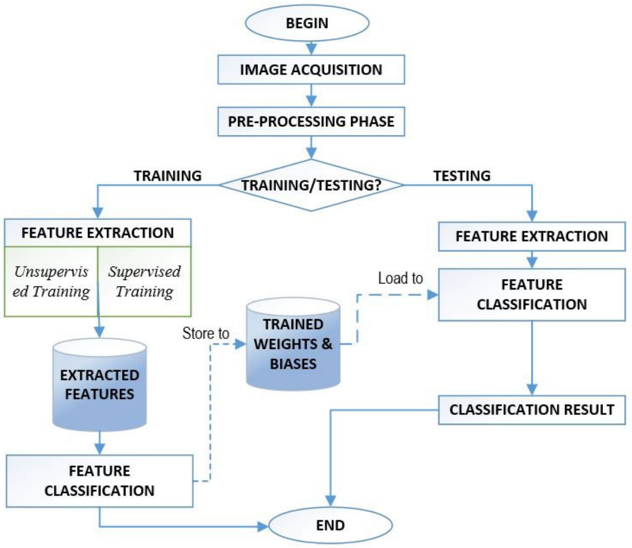


FIGURE 2. The workflow of the proposed technique.

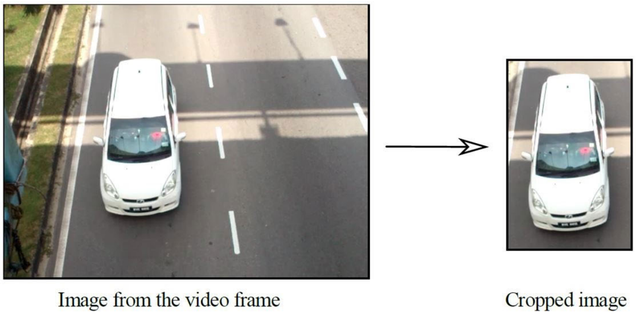


FIGURE 3. Example of image from the video frame and the cropped image.

the feature classification phase. In the end, the classification result is obtained. The detail explanation about each process is provided in the following sections:

A. IMAGE ACQUISITION

Videos which contain vehicles images are recorded using a surveillance camera. A region in the video frame containing a vehicle is manually cropped to ensure the classification process is focused on the vehicle.

Figure 3 shows the illustration of the cropping procedure. The cropped image size and aspect ratio is varied. It is due to the vehicle size variations. Later, the cropped image is stored into a training and a testing dataset.

B. PRE-PROCESSING PHASE

There are two processes in the pre-processing phase, the first process is to resize the image with maintained aspect ratio. The second process is to convert the RGB color space image to the YCrCb color space image. Figure 4 depicts the processes involved in the pre-processing phase. The first process is implemented by fetching an input image from a dataset, and resize it while maintaining the aspect ratio. Thus, the shape of the vehicle remains consistent by implementing this process.

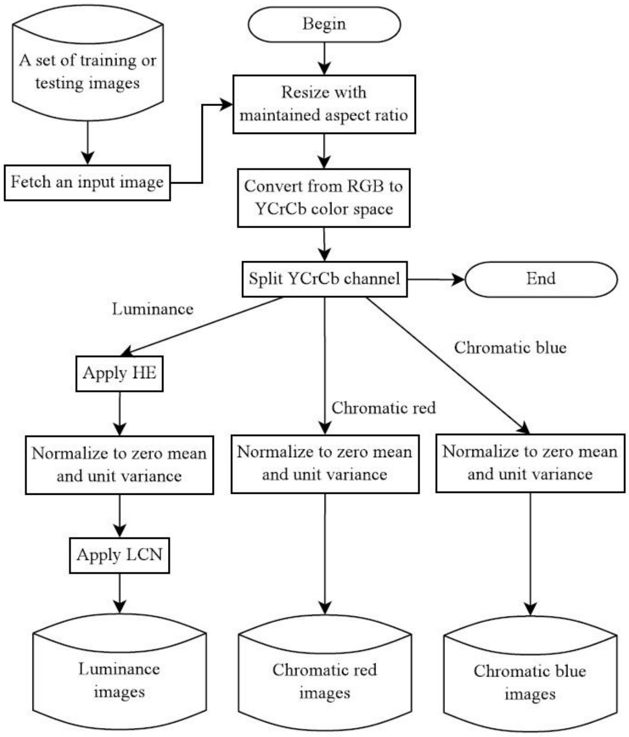

A set of training or testing images
Begin
Resize with maintained aspect ratio
Fetch an input image
Convert from RGB to YCrCb color space
Split YCrCb channel
End
Luminance
Chromatic blue
Apply HE
Chromatic red
Normalize to zero mean and unit variance
Normalize to zero mean and unit variance
Normalize to zero mean and unit variance
Apply LCN
Luminance images
Chromatic red images
Chromatic blue images

FIGURE 4. The process flow in the pre-processing phase.

After that, the image that is initially in RGB color space is then converted to YCrCb color space. The conversion process is done based on the standard conversion formula in the study done by Bautista *et al.* [11]. Next, the YCrCb channel image is split apart into three components. The advantages of using YCrCb instead of RGB are Y is computed from nonlinear RGB, Cr is storing a difference between red and luma component, and Cb is storing a difference between blue and luma component.

Also, YCrCb color space is luminance independent, thus, it will give a better performance compared to RGB [11]. In contrast to RGB, YCrCb has separate components of luminance and chrominance that make it attractive to color image segmentation and feature extraction.

The next processes in the pre-processing phase are histogram equalization (HE), normalizing to zero-mean and unit variance, and local contrast normalization (LCN) that will be implemented on each image. HE process is implemented on the luminance channel image followed by normalization to zero-mean, and unit variance process on the luminance image, the chromatic red image, and the chromatic blue image. The LCN process is implemented in the luminance image. The purpose of implementing the LCN process is to remove shading and illumination light to enhance the vehicle component in the luminance channel. In the end, the outputs from this phase are a set of luminance, chromatic red, and chromatic blue images. These outputs will be used as an input to the feature extraction process.However, HE and LCN will not be implemented to the chromatic channel images to protect the color information in this pre-processing process [17].

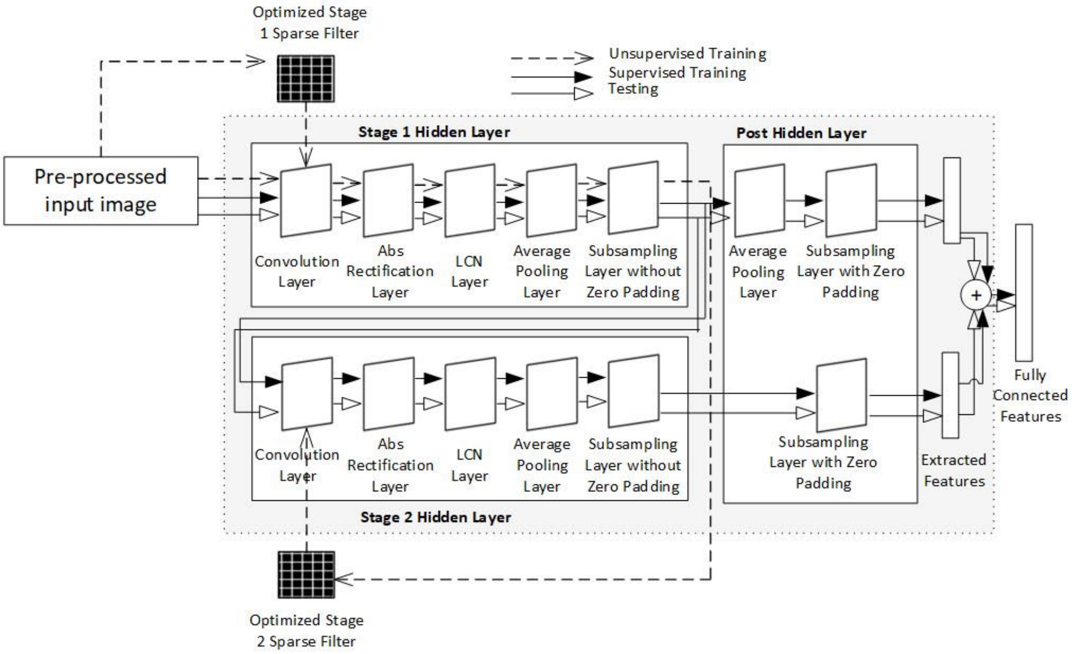


FIGURE 5. Overview of SF-CNNLS Technique.

C. FEATURE EXTRACTION PHASE

This section will explain how to extract the vehicle features from the pre-processed image. The features are extracted based on the processes in SF-CNNLS components, as shown in figure 5. SF-CNNLS has two hidden layers and one post hidden layer. The reason why we use two hidden layers is to extract both local and global features of vehicles. Therefore, for the first hidden layer, the aim is to extract local features of a vehicle based on the optimized Sparse Filter that has been generated from the pre-processed input image. The second hidden layer is implemented to extract global features of a vehicle based on the optimized Sparse Filter that generated from the extracted features from the first hidden layer. Hence, the post hidden layer is determined to process the extracted features from the previous hidden layers before vectorizing the features into a one-dimensional vector. In this section, the process of the feature extraction in the training phase will be explained. As mentioned before, the training phase consists of unsupervised and supervised training. Features of the images from the training dataset are extracted in these two pieces of training. Therefore, the feature extraction phase is explained based on the processes involved in the unsupervised and supervised training as follows:

- **Unsupervised Training:** Unsupervised Training aims to extract local features (e.g., point, edges, and corner of the vehicle) from input images, and to generate optimized sparse filters. Initially, input images which is the pre-processed images are delivered into sparse filtering function. This process aims to generate optimized stage 1 sparse filters. The optimized stage 1 sparse filters are required to handle and learn the high-dimensional local features. Therefore, the sparse filtering function is used to generate the filters. The reason why we used this function is because it has been tested as the fastest filtering function in unsupervised training [17]. Once the filters are generated, the pre-processed images are delivered to the SF-CNNLS stage 1 hidden layer. After that, a set of optimized stage 2 sparse filters is generated by the sparse filtering function at stage 2 hidden layer. These filters are generated using the produced output at the SF-CNNLS stage 1 hidden layer. We will explain the process of generating the optimized sparse filters prior the explanation of each component is inside the layer.
  Note that SF-CNNLS needs trainable filters to extract the features. In this study, sparse filtering function is implemented with several steps. Figure 6 shows the process flow of the sparse filtering function. The steps are implemented in two main phases, which are the pre-processing phase, and the optimum sparse filters production phase. The pre-processing is implemented on the pre-processed image prior to the optimum sparse filters are produced. This pre-processing has two sub-processes. The sub-processes split the pre-processed training images into small patches, and normalize the patches by applying zero-mean and unit variance normalization on each patch. The filtering function begins with initializing the sparse filters with random normal distribution number. The next process is vectorizing the input patches, and after that Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimization algorithm is configured.
  The BFGS optimization algorithm is used to minimize the back-propagation objective function in order to produce the optimized sparse filters. The BFGS implementation is based on a GNU GSL package. Several tests are conducted to get the optimum convergence in determining the acceptable number of iteration.

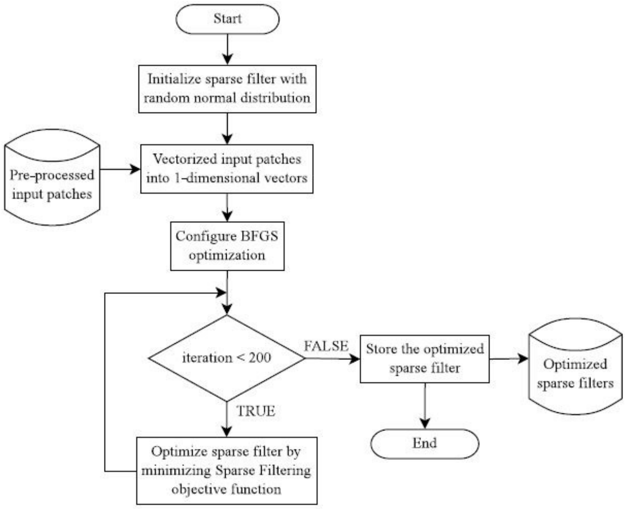

Start
Initialize sparse filter with random normal distribution
Pre-processed input patches
Vectorized input patches into 1-dimensional vectors
Configure BFGS optimization
iteration < 200
FALSE
TRUE
Store the optimized sparse filter
Optimized sparse filters
End
Optimize sparse filter by minimizing Sparse Filtering objective function

FIGURE 6. Process flow of sparse filtering function.

Based on the tests, the optimum convergence is achieved when the iteration of 200 is met. After that, each sparse filter is normalized into [−1, 1]. It is to avoid exponent overflow in the next processes. Overall, there are 64 optimized sparse filters with the size of 9x9 each are produced from each channel in stage 1. Another 9 optimized sparse filters with the size of 9x9 each are produced from stage 2.

After that, the optimized stage 1 sparse filters are convolved with the pre-processed input images to extract the features. The sigmoid activation function is implemented subsequently to each of the convolved images using equation (1) and (2). The output from this layer is the extracted features that have a smaller size than the input size. For instance, let the size of the input image is 256 x 174 pixels, and convolved with 64 optimized filters, therefore, the output is 64 local features.

$$xk_i = sig(x \otimes f_i) \tag{1}$$

$$sig(x) = \frac{1}{1+\exp(-x)} \tag{2}$$

where xk_i is optimized sparse filters, x is the pre-processed image as an input image, $f_i$is the extracted features and $sig(.)$ is sigmoid activation function.

Next, the Absolute Value Rectification (AVR) layer is implemented. In this layer, the absolute value operation is applied to the extracted features that have been produced from the previous component. Thus, the output from this layer have absolute value elements and will be an input to the next component.

Next, the LCN is applied to the input. Subtractive and divisive operations in this LCN is similar to the LCN in the pre-processing phase. However, the maximum value, and the input in here are from the output of the AVR layer [16]. Thus, the output here is a set of LCN normalized features. The output has the same size as the input image and will be used in the average pooling layer.

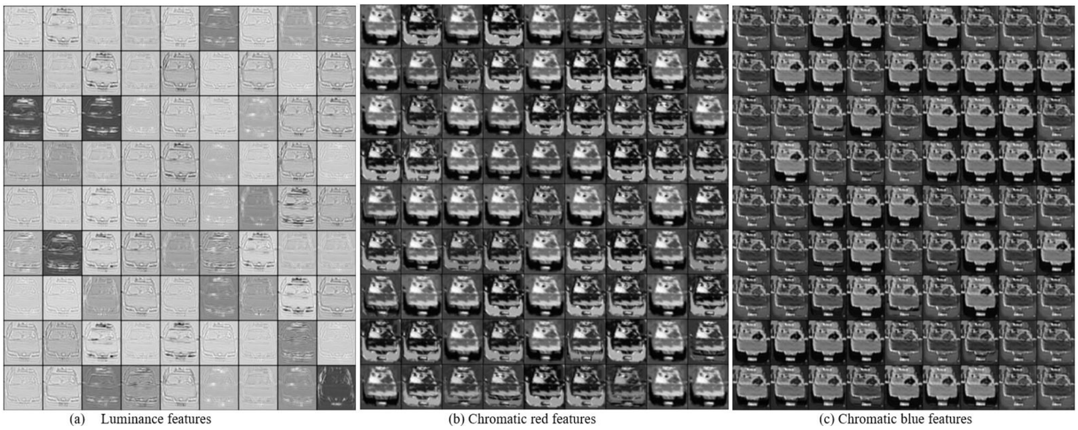
Then, the average pooling layer component is implemented. This component is implemented to make the extracted features less sensitive to variation in angle and size of a vehicle. Hence, it is able to enhance the competition amongst features. Average pooling does not reject all of the features and retains more information of the vehicles. Here, the input which is the extracted features from the previous component will be convolved with the average filtering. The objective of doing this is to make the features become less sensitive to the variation of angles and size of a vehicle. This layer also will reduce the sensitivity towards geometric distortion. The subsampling layer without zero padding is the last component inside the stage 1 hidden layer. The process is similar to the resize with a maintained aspect ratio as implemented in the pre-processing phase except that the input here is the extracted features from the average pooling layer. As a result, the extracted features are produced at the end of the stage 1 hidden layer for the unsupervised training. The extracted features will be used to generate the optimized stage 2 sparse filters in the stage 2 hidden layer.

- **Supervised Training:** Supervised Training aims to extract local features of a vehicle at stage 1 hidden layer, and global features (e.g., contour, size, and shape of the vehicle) at stage 2 hidden layer. Similar processes of extracting the vehicle features as in the unsupervised training are implemented during the supervised training, as shown in figure 4. The input for this training is the extracted features that have been generated at stage 1 hidden layer.

  Initially, the optimized stage 2 sparse filters will be convolved with the input at the first component of stage 2 hidden layer. In this stage, the optimized sparse filter is trainable filters from the local features that have been extracted in the hidden layer stage 1, as explained in the unsupervised training. All the components inside the stage 2 hidden layer are deployed accordingly with the same process as in stage 1 hidden layer.

  Note that the extracted local and global features from both hidden layers are inputs to the post hidden layer. The extracted local features from stage 1 hidden layer are processed by the average pooling and the subsampling layer with zero padding. At the same time, the extracted global features from stage 2 hidden layer are processed by the subsampling layer with zero padding. After that, the output from the post hidden layer will be concatenated at the fully connected component. The concatenated features from each component will be fully connected prior to the classification process.

  Overall, the extracted features are gathered from three components of the image channels, as mentioned earlier. Figure 7 shows the example of the extracted features of a taxi. In that figure, the luminance features consist of local and global features that have been extracted from



(a) Luminance features

(b) Chromatic red features

(c) Chromatic blue features

FIGURE 7. Example of extracted features of a taxi in three channel of image.

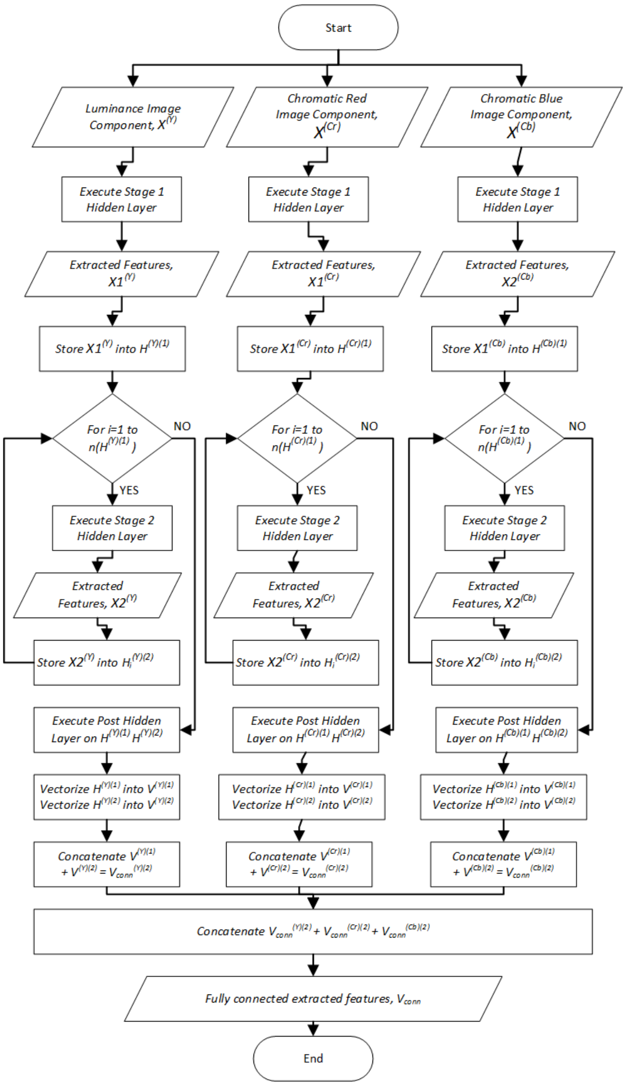

Start
Luminance Image Component, $X^{(Y)}$
Chromatic Red Image Component, $X^{(Cr)}$
Chromatic Blue Image Component, $X^{(Cb)}$
Execute Stage 1 Hidden Layer
Execute Stage 1 Hidden Layer
Execute Stage 1 Hidden Layer
Extracted Features, $X1^{(Y)}$
Extracted Features, $X1^{(Cr)}$
Extracted Features, $X2^{(Cb)}$
Store $X1^{(Y)}$ into $H^{(Y)(1)}$
Store $X1^{(Cr)}$ into $H^{(Cr)(1)}$
Store $X1^{(Cb)}$ into $H^{(Cb)(1)}$
For i=1 to $n(H^{(Y)(1)})$
NO
YES
For i=1 to $n(H^{(Cr)(1)})$
NO
YES
For i=1 to $n(H^{(Cb)(1)})$
NO
YES
Execute Stage 2 Hidden Layer
Execute Stage 2 Hidden Layer
Execute Stage 2 Hidden Layer
Extracted Features, $X2^{(Y)}$
Extracted Features, $X2^{(Cr)}$
Extracted Features, $X2^{(Cb)}$
Store $X2^{(Y)}$ into $H_i^{(Y)(2)}$
Store $X2^{(Cr)}$ into $H_i^{(Cr)(2)}$
Store $X2^{(Cb)}$ into $H_i^{(Cb)(2)}$
Execute Post Hidden Layer on $H^{(Y)(1)}$ $H^{(Y)(2)}$
Execute Post Hidden Layer on $H^{(Cr)(1)}$ $H^{(Cr)(2)}$
Execute Post Hidden Layer on $H^{(Cb)(1)}$ $H^{(Cb)(2)}$
Vectorize $H^{(Y)(1)}$ into $V^{(Y)(1)}$ Vectorize $H^{(Y)(2)}$ into $V^{(Y)(2)}$
Vectorize $H^{(Cr)(1)}$ into $V^{(Cr)(1)}$ Vectorize $H^{(Cr)(2)}$ into $V^{(Cr)(2)}$
Vectorize $H^{(Cb)(1)}$ into $V^{(Cb)(1)}$ Vectorize $H^{(Cb)(2)}$ into $V^{(Cb)(2)}$
Concatenate $V^{(Y)(1)} + V^{(Y)(2)} = V_{conn}^{(Y)(2)}$
Concatenate $V^{(Cr)(1)} + V^{(Cr)(2)} = V_{conn}^{(Cr)(2)}$
Concatenate $V^{(Cb)(1)} + V^{(Cb)(2)} = V_{conn}^{(Cb)(2)}$
Concatenate $V_{conn}^{(Y)(2)} + V_{conn}^{(Cr)(2)} + V_{conn}^{(Cb)(2)}$
Fully connected extracted features, V_{conn}
End

FIGURE 8. Process flow of the feature extraction algorithm for 3 channels of image using TC-SF-CNNLS.

Y component. The features are almost similar when we extracted from a grayscale image. Whereas, the local and global features of the taxi have been extracted from the chromatic red and blue component, respectively. Notice that the local and global features in the chromatic red and blue are richer compared to the luminance features. The unique features of the vehicle, for instance, the taxi's sign is clearer compared to the local features that we obtained in the luminance image.

Therefore, there is richer information about local and global features are extracted to be classified. Figure 8 shows the process flow of the algorithm that we implemented for all the features in each component. Based on that flow, the inputs are luminance image, chromatic red image, and chromatic blue image. These inputs are processed to extract low-level and high-level features one by one. The reason we use this algorithm is that we have transformed the original input image into three channels of the image. Therefore, there will be three processes to extract the vehicle features from each of the image component. The process is started with stage 1 hidden layer that is executed as shown in figure 3 to extract the low-level features. After that, stage 2 hidden layer is executed for each of the extracted features to extract the high-level features. Once the execution of stage 2 hidden layer is completed, the low-level and high-level of extracted features are being processed in the post hidden layer. In the end, the output of that algorithm is a fully connected 1-dimensional features vector contains the extracted features. The extracted features consist of low-level and high-level features from luminance, chromatic red and chromatic blue image.

- **Testing:** Testing dataset consists of the pre-processed images will be extracted the features in the feature extraction phase as shown in figure 4. The feature extraction processes are similar to the feature extraction processes in the supervised training phase. The output is the extracted local and global features that will be concatenated in the fully connected features. The concatenated features will be used in the classifier to calculate classification probabilistic in obtaining classification results. The further processes of the testing phase will be explained in the classification phase.

D. FEATURE CLASSIFICATION PHASE

Softmax Regression is used as a classifier in the feature classification phase. The extracted features from the feature extraction phase are utilized in this classifier to classify the vehicle class. Classifying the vehicle is done by executing a standard calculation of Softmax Regression hypothesis in equation (3), as shown at the bottom of this page, where, let $h_\theta \in \Re^e$ is hypothesis for N vehicle classes. $x \in \Re^e$ is the extracted vehicle features, e is number of elements in a vector x, $\theta_n \in \Re^e$ is weight for each class, $d_n \in \Re^e$ is bias for each class, q is actual result and $HP_t q(1 \mid x; \theta, d)$ is probability of $q = n$ when feature x is given.

For instance, the Softmax Regression is trained to recognize $N = 3$ types of vehicle class. The types of class are car, taxi and truck, whereby, $n = 1, 2, 3$ represents car class, taxi class, and truck class, respectively. Thus, 1-dimensional vector of the extracted features denoted as x with a length of e. Softmax Regression will have N number of trained weights, θ_n, and N number of trained biases, d_n since it is trained with N classes. The hypothesis,h_θ, for N vehicle classes is 1-dimensional vector with a length of N. h_θ contains a list of N probabilities for the vehicle classes. Therefore, to identify n class that has highest probability, the probability of actual result q equals to n class denoted as $HP_t = q = n \mid x; \theta, d$ is calculated. Later, the calculated probability is divided by $\sum_{j=1}^{N} \exp(\theta_j^T x + d_n)$ to form it in [0.0-1.0]. Thus, only one n with the highest probability is selected. For example, the hypothesis result is $h\theta = \begin{bmatrix} 0.1013 \\ 0.9203 \\ 0.0455 \end{bmatrix}$ therefore, the feature x is classified as taxi class due to the second index of the hypothesis has the highest probability, and the index is $n = 2$ represents the taxi class. The explained classification using the classifier is implemented based on the training phase, which is the supervised training, and the testing phase. The classification process in each phase is explained as follows:

- **Supervised Training:** Supervised Training trained Softmax Regression to produce trained weights and biases in the supervised training. The training process is implemented by minimizing both negative log-likelihood as in equation (4), as shown at the bottom of this page, and Mean Squared Error (MSE) using a gradient descent method as in equation (5), as shown at the bottom of this page, The process of minimizing the negative log-likelihood equation is similar to the process of minimizing the Kullback–Leibler divergence [7], [8]. Equation (6) and (7), as shown at the bottom of this page, are derivative equations for updating weight and bias.
  where N is a number of vehicle classes, XT is training features from N. $x \in \Re^e, n = 1, 2, 3, \ldots K$ is the extracted features, and e is a number of elements in x. $1\{q_i = n\}$ is a function that returns 1 if q_i is equal to n, otherwise 0. $\theta_n \in \Re^e$, $n = 1, 2, 3, \ldots N$ is weight and $d_n \in \Re^e$ with $n = 1, 2, 3, \ldots N$ is biases. q is the actual result and λ is a non-negative regularization parameter. The equations (4) and (5) are calculated to minimize the output of the negative log-likelihood. The aim of these calculations is to update the θ_n and the d_n iteratively. The MSE equation is calculated for each end of the iteration. It is to monitor the progress of the supervised training. A non-negative regularization parameter for the Softmax Regression is denoted as λ to control the generalization performance. The trained weights and biases from this

$$h_\theta = \begin{bmatrix} HP_t q(1 \mid x; \theta, d) \\ HP_t q(2 \mid x; \theta, d) \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ HP_t q(K \mid x; \theta, d) \end{bmatrix} + \frac{1}{\sum_{j=1}^{K} \exp(\theta_j^T x + d_n)} \begin{bmatrix} \exp(\theta_1^T x + d_1) \\ \exp(\theta_2^T x + d_2) \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \exp(\theta_n^T x + d_n) \end{bmatrix} \tag{3}$$

$$L(\theta) = -\frac{1}{XT}\left[\sum_{i=1}^{XT}\sum_{n=1}^{N} 1\{q_i = n\} \log HP_t(q_i = n \mid x_i, \theta, d)\right] + \frac{\lambda}{2}[\![\theta]\!]_2^2 \tag{4}$$

$$MSE = \frac{1}{N \bullet XT}\left[\sum_{n=1}^{N}\sum_{i=1}^{XT} HP_t(q_i = n \mid x_i, \theta, d) - 1\{q_i = n\}\right]^2 \tag{5}$$

$$\underset{\theta_n}{\nabla} L(\theta) = -\frac{1}{XT}\sum_{i=1}^{XT}[x_i(1\{q_i = n\} - HP_t(q_i = n \mid x_i, \theta, d))] + \lambda\theta_n \tag{6}$$

$$\underset{d_n}{\nabla} L(\theta) = -\frac{1}{XT}\sum_{i=1}^{XT}[1\{q_i = n\} - HP_t(q_i = n \mid x_i, \theta, d)] \tag{7}$$

$$HP_t(q_i = n \mid x_i, \theta, d) = \frac{\exp(\theta^{(n)T} x^{(i)} + d^{(n)})}{\sum_{i=1}^{N} \exp(\theta^{(j)T} x^{(i)} + d^{(j)})} \tag{8}$$

TABLE 1. Confusion Matrix, Accuracy, Precision, Recall and F-Score: TC-SF-CNNLS vs SF-CNNLS (3 Classes).

Technique		TC-SF-CNNLS							SF-CNNLS [17]						
Class		Actual			Accuracy	Precision	Recall	F-score	Actual			Accuracy	Precision	Recall	F-score
		Car	Taxi	Truck					Car	Taxi	Truck				
Predicted	Sedan Car	195	3	2	0.9833	0.9750	0.9750	0.9750	121	75	15	0.5617	0.5735	0.6050	0.5888
	Taxi	4	197	0		0.9801	0.9850	0.9825	69	91	60		0.4136	0.4550	0.4333
	Truck	1	0	198		0.9950	0.9900	0.9925	10	34	125		0.7396	0.6250	0.6775

phase will be stored in a database to be loaded into the classification phase in the testing phase.

- **Testing:** Testing procedure is started with the trained weights and biases are loaded from the database and the hypothesis in calculated for each class using equation (3). The aim of this procedure is to obtain results of the classification performance. The results are measured based on predicted and actual classes in a confusion matrix to observe an accuracy, a precision, a recall and an f-score of the proposed technique. For the multi classes of vehicle, N_j, there will be tp_i is a true positive, fp_i is false positive, fn_i is a false negative, and tn_i is a true negative. Thus, equations (9) to (12) below are used to calculate the results.

$$Accuracy = \sum_{i=1}^{N} \frac{tp_i + tn_i}{tp_i + fn_i + fp_i + tn_i} \tag{9}$$

$$Precission = \frac{\sum_{i=1}^{N} tp_i}{\sum_{i=1}^{N}(tp_i + fp_i)} \tag{10}$$

$$Recall = \frac{\sum_{i=1}^{N} tp_i}{\sum_{i=1}^{N} tp_i + fn_i)} \tag{11}$$

$$F\!-\!score = \frac{2 * Precission * Recall}{Precission + Recall} \tag{12}$$

IV. EXPERIMENTS AND RESULTS

A. DATASET AND SETTINGS

The experiment is conducted using a benchmark database known as BIT vehicle dataset [7]. This database is selected because it consists of variety of vehicle classes, for instance, bus, car (passenger car), minivan, SUV, taxi, and truck in which, other databases are not providing taxi images. Furthermore, the angles of the vehicle are the top and frontal view that suitable for our future system's implementation, and the images are captured during daylight with different illumination condition from a traffic surveillance mounted-camera. However, the dataset does not provide a specific time when were the images are captured. Figure 9 shows the example of vehicle images. The total number of vehicle images in this dataset is 9850 images. Car and SUV have the highest number of images with approximately 5000 and 1300 images, respectively. Taxi and minivan have the lowest number of images with not more than 600 images each. Thus, to avoid bias during the training and testing phases, 250 images are randomly selected from each class as the training dataset, and 200 images as the testing dataset. For the parameter setup during the experiment, the parameter value for training rate is 0.001. The momentum rate is 0.9 and regularization is 0.2, 0.4, 0.6, 0.8, 1.0 and 2.0 for each test to observe which regularization value that will produce the best result. The results are observed based on two major experiments. The first experiment is three classes of vehicle, and the second experiment is six classes of vehicle. Next subsections consist of the results that we obtained based on the testing dataset.

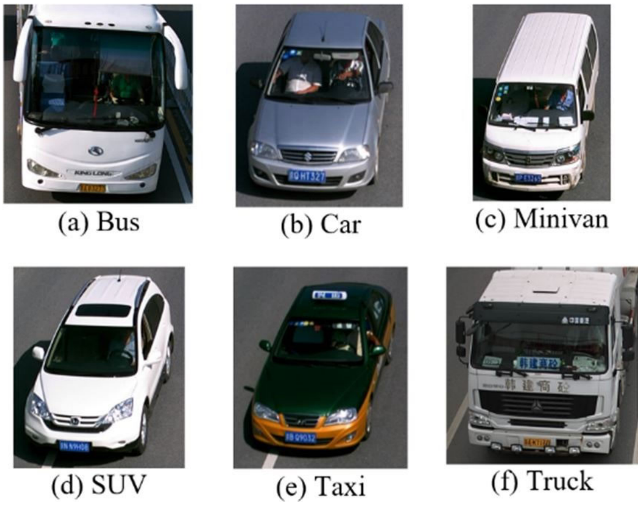

(a) Bus (b) Car (c) Minivan (d) SUV (e) Taxi (f) Truck

FIGURE 9. Vehicle images in BIT dataset.

B. RESULTS FOR 3 CLASSES

The performance of TC-SF-CNNLS for three classes is evaluated based on sedan car, taxi and truck vehicle type to compare it with the performance of the existing technique of SF-CNNLS [17]. In total, there are 750 images for the training dataset and 600 images for the testing dataset. For sedan car class, all types of the sedan car that consists of the sedan car without sunroof and with sunroof are utilized. Table 1 shows the obtained results in a confusion matrix for both techniques. The table consists of an average accuracy, precision, recall and f-score for each technique. The results represent the highest performance achieved when the regularization value is 1.0 for both techniques.

Looking at table 1, the average accuracy of TC-SF-CNNLS is 0.9833 compared to SF-CNNLS with only 0.5617. The precision of TC-SF-CNNLS in each class is more than 0.995 compared to the highest precision in SF-CNNLS with only 0.7396. It shows that TC-SF-CNNLS has a low false positive rate compared to the later technique. Other than that, the recalls and f-scores in TC-SF-CNNLS are higher than in SF-CNNLS. The results show that the rate of correctly predicted in each actual class is high when we use TC-SF-CNNLS.

TABLE 2. Confusion Matrix, Accuracy, Precision, Recall and F-Score: TC-SF-CNNLS (6 Classes).

Class		Actual						Accuracy	Precision	Recall	F-score
		Bus	Car	Minivan	SUVT	Taxi	Truc				
Predicted	Bus	186	0	0	0	0	18	0.9041	0.9118	0.9300	0.9208
	Car	0	190	7	6	12	0		0.8837	0.9500	0.9156
	Minivan	0	6	171	24	0	2		0.8424	0.8550	0.8486
	SUVT	0	2	22	170	0	0		0.8762	0.8500	0.8629
	Taxi	0	2	0	0	188	0		0.9895	0.9399	0.9641
	Truc	14	0	0	0	0	180		0.9278	0.9000	0.9137

Note that, in SF-CNNLS, the sedan car with sunroof gives a low performance. It is due to the features of the sunroof share almost similar features as the taxi's sign when appeared on an image. Thus, one of the aims in this experiment is to observe if the proposed technique able to give a better performance when dealing with this vehicle features. Based on the results, we can see that TC-SF-CNNLS able to correctly recognized all the vehicle classes especially the taxi class. Overall, we can conclude that our proposed TC-SF-CNNLS outperforms the existing technique in this experiment.

C. RESULTS FOR 6 CLASSES

We test our proposed technique with more vehicle classes to observe the technique performance when the determined vehicle classes are involved. There are six vehicle classes considered in this experiment. The classes are the bus, sedan car, minivan, SUV, taxi and truck. We presented the results in table 2.

We select the vehicle images randomly in each class, whereby, 250 images are for training dataset and 200 images for testing dataset. The total training dataset is 1500 images and 1200 images in the testing dataset. Similar to the previous experiment, various types of sedan car are used, including cars with sunroof. Table 2 consists of the highest results when the regularization value is 1.0.

Looking at table 2, the average accuracy is 0.905. Whereas, the highest precision is from taxi class with 0.9548, and the lowest precision is from the minivan class with 0.8629. Based on these results, the classification is considered precise because the proportion of the predicted positives to the actual positive are consistently high in all classes. Other than that, the recalls and f-score are higher with more than 0.83 and 0.8564, respectively. It shows that the rate of correctly predicted in each actual class is high. From this experiment, we can see that this proposed technique has a promising performance because the accuracy, precision, recall, and f-score are paralleled consistent.

We also compare the results with the existing techniques, which are, SF-CNNLS [17], and semi-supervised CNN [10]. Semi-supervised CNN [10] is selected as one of the techniques to be compared because this technique has an almost similar framework as our proposed technique and it achieved the best accuracy performance as reported in the related works. Furthermore, they used the same database which is BIT in their experiment. Thus, this technique is suitable to be compared in this experiment. The existing techniques are re-implemented and tested to obtain the results that valid to be compared. In addition, a taxi is classified as a sedan car class in semi-supervised CNN [10].

A comparison graph in figure 10 shows the results performance for each technique. Based on that figure, it is clearly seen that TC-SF-CNNLS able to outperform other two techniques in all results performance in terms of the average accuracy, precision, recall and f-score. The average accuracy of TC-SF-CNNLS achieves 0.90, which surpasses semi-supervised and SF-CNNLS with not more than 0.79 and 0.60, respectively. Whereas, TC-SF-CNNLS precision is higher than the other two techniques in all the vehicle classes with between 0.85 to 0.96, similar to the recall and f-score with 0.80 to 0.96, and 0.85 to 0.95, respectively. Therefore, we can see that the proposed technique is more accurate, precise and less sensitive compared to the other two techniques.

D. RESULTS FOR SELF-OBTAINED DATASET (SPINT)

In this experiment, TC-SF-CNNLS is tested with a self-obtained dataset named as SPINT. This dataset is obtained in Malaysia federal road during daylight within 1 week from 10am to 1pm, and 3pm to 6pm. The reason why we test with this dataset is to measure the efficiency of this technique when we test with the local vehicles. This is because the local vehicles have different features compared to the benchmark vehicles in certain vehicle classes. The outcome of this experiment can be a significant input for further improvement of this technique in order to implement it in Malaysia transportation systems.

Figure 11 shows the example of vehicle images in this dataset. There are 3500 vehicle images in total with 6 classes. The classes are car or sedan car, taxi, minivan, SUV, bus and truck. Car or sedan car, SUV, and minivan have the highest number of images. Whereas, taxi and bus have the lowest number of images which is not more than 400 images. We randomly select 200 images from each class for the training dataset and 200 images as the testing dataset.

Table 3 consists of the highest average accuracy, precision, recall, and f-score that are obtained from this experiment. The highest average accuracy is 0.9041. Whereas the highest precision is from the taxi class with 0.9895, and the lowest precision is from the minivan class with 0.8424.The recall rate is between 0.85 to 0.95, which shows that the actual vehicle class that correctly identified is high, as well as the f-score values between 0.8486 to 0.9641. Based on these results,

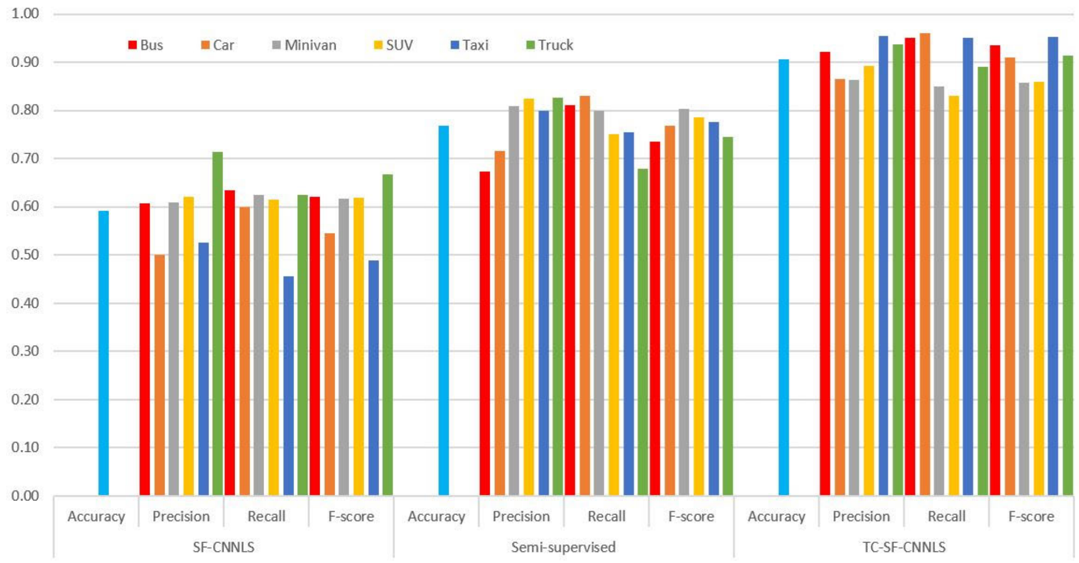


FIGURE 10. Performance comparison between 3 Techniques.

TABLE 3. Confusion Matrix, Accuracy, Precision, Recall and F-Score: TC-SF-CNNLS (6 Classes) Using Spint Data set.

Class		Actual						Accuracy	Precision	Recall	F-score
		Bus	Car	Minivan	SUVT	Taxi	Truc				
Predicted	Bus	186	0	0	0	0	18	0.9041	0.9118	0.9300	0.9208
	Car	0	190	7	6	12	0		0.8837	0.9500	0.9156
	Minivan	0	6	171	24	0	2		0.8424	0.8550	0.8486
	SUVT	0	2	22	170	0	0		0.8762	0.8500	0.8629
	Taxi	0	2	0	0	188	0		0.9895	0.9399	0.9641
	Truc	14	0	0	0	0	180		0.9278	0.9000	0.9137

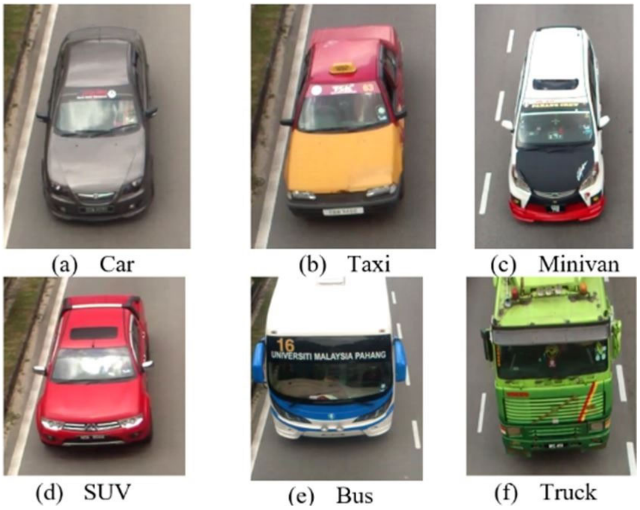


FIGURE 11. Vehicle images in SPINT dataset.

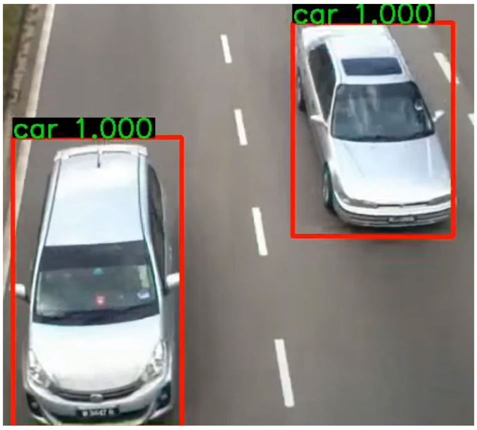


FIGURE 12. Example of the system implementation.

the classification using this dataset is considered accurate, precise and less sensitive in all classes. However, the results are slightly low compared to when we test with the BIT dataset. This is due to the quality of our images and the number of samples that lower than BIT samples in the training dataset. Figure 12 is an example of the system implementation based on this proposed technique. The system is implemented for vehicle census in Malaysia. In the figure, the label of ''car 1.000'' indicates that the vehicle is 100% recognized as car class.

We also observe the executing time based on the video that we obtained. Based on the observation, the average time to classify one vehicle is 5.65 seconds. It is reasonably efficient in terms of the classification process when the position of the camera and the distance of the camera to the region of interest are considered. However, it can be improved in the future to enhance the executing time.

TABLE 4. Comparison of accuracy performance with other techniques.

Class	Technique				
	Dong et al. [10]	Bautista et al. [11]	Chen et al. [24]	Li et al. [23]	Proposed Technique
Bus	98%	96.47%	97.8%	95.04%	98.5%
Sedan Car	91%				98.5%
Truck	90%				97.0%
Minivan	83%		-	-	86.0%
SUV	84%	-	-	-	89.0%

E. RESULTS COMPARISON WITH OTHER STATE-OF-ART TECHNIQUES

In this experiment, TC-SF-CNNLS is tested with similar vehicle classes as reported in the state-of-art techniques using BIT database. The vehicle classes that we observed are bus, sedan car, minivan, SUV and truck. In this experiment, we consider taxi as sedan car class. This is because all the related works did not separate the taxi as another class. The techniques that we considered in this comparison are Semi-supervised CNN [10], CNN in low-resolution images [11], multi-branch CNN [23], and compress sensing and CNN [24]. The reason why we choose these techniques is that they used CNN based framework in their work. Therefore, we would like to see if this proposed technique is comparable to other related techniques. For this comparison, we randomly select 200 images for the training dataset and 200 images for testing dataset similar to the experiment in work done by Dong *et al.* [10]. Table 4 shows the comparison of the results among those techniques.

Looking at that table, only Dong *et al.* [10] provided the highest accuracy for each vehicle type, whereby, bus class achieved the highest of 98% accuracy, followed by sedan car with 91%, truck, minivan and SUV with 90%, 83%, and 84%, respectively. As for Bautista *et al.* [11], they used low-resolution images taken from surveillance cameras in their experiment. However, they only classified into bus, car, truck, and minivan. They were able to achieve an average accuracy of 96.47%. Whereas, Chen *et al.* [23] only provided the highest accuracy that they obtained without detailed accuracy in each class. They tested with three databases created by themselves. The databases contain web-nature images, a multi-view angle (no top view angle), and the images are taken in a controlled environment such as in a showroom. Based on their experiment, the highest accuracy for each database is 91%, 94.9%, and 97.8%. However, they did not mention in detail the specific vehicle class that they observed in their experiment, except general vehicle classes, for instance, bus, car, and truck.

Another technique is proposed by Li *et al.* [24]. In this work, they only observed their accuracy performance based on 3 vehicle classes, which are a bus, sedan car and truck using Caltech database with the average accuracy is 95.04% without a specific accuracy in each class. In our proposed technique, we observe the accuracy performance in each class. The highest accuracies are obtained from bus and sedan car with an accuracy of 98.5%. Truck, minivan, and SUV achieve the accuracy of 97%, 86% and 89%, respectively. All the accuracies are higher than the accuracies in all classes from Dong *et al.* [10]. From this comparison, we can see that our proposed method is comparable and outperforms other techniques with the highest accuracy is 98.51%.

Thus, this proved that our proposed technique enhances the technique of unsupervised and supervised CNN by utilizing the local and global features from the luminance and the chromatic component of vehicle image is able to successfully differentiate the vehicle classes. It also proves that the vehicle classes are able to be classified based on the determined classes.

V. SUMMARY AND CONCLUSION

A vehicle type classification has been proposed using an enhanced technique named three-channels of a convolutional neural network with a layer-skipping strategy (TC-SF-CNNLS) to overcome the difficulties in classifying the vehicle from different classes that shared almost similar features. The aim of this study is to extract unique and richer features of the vehicle with the hypothesis that the features able to differentiate the almost similar features. The technique mimics the ability of human eyes that have different sensitivity to color and brightness when looking at an object. Due to that, this technique is designed to extract the local and global features of an object which is a vehicle from luminance and chromatic components of the images. The extracted features are separated in each component, thus, we are able to extract the discriminant features to obtain unique features among the vehicles. The luminance component denotes as Y, which is a grayscale color space. This component represents brightness in human eyes. It is used to extract the local and global features of the vehicle. The chromatic red component and chromatic blue component denote as Cr and Cb, respectively. Cr corresponds to reddish colors and Cb corresponds to blueish colors. In general, Cr and Cb represent the color information of the image.

In this technique, the sparse filtering is optimized to generate the optimized sparse filters that are significant in extracting the features from three components. Also, supervised training and unsupervised training are implemented to extract and learn the features. Softmax Regression is used as the classifier to produce an output probability of the vehicle. A benchmark database is known as the BIT dataset, and a self-obtained dataset known as SPINT that consists of 6 vehicle classes is used to observe the performance of this proposed technique. Based on the experimental results, TC-SF-CNNLS is able to successfully differentiate vehicle classes especially the class that has similar features, such as taxi and car, minivan and SUV. Thus, it is proved that the proposed technique is able to produce a promising classification accuracy compared to the other existing techniques.

REFERENCES

[1] S. Awang and N. M. A. N. Azmi, "Vehicle counting system based on vehicle type classification using deep learning method," in *IT Convergence and Security*. Singapore: Springer, 2017, pp. 52–59.

[2] S. Awang and N. M. A. N. Azmi, ''Automated toll collection system based on vehicle type classification using sparse–filtered convolutional neural networks with layer–skipping strategy (SF–CNNLS),'' *J. Phys., Conf.*, vol. 1061, Jul. 2018, Art. no. 012009.

[3] H. Jiang, H. Li, T. Liu, P. Zhang, and J. Lu, ''A fast method for RGB to YCrCb conversion based on FPGA,'' in *Proc. 3rd Int. Conf. Comput. Sci. Netw. Technol.*, Oct. 2013, doi: 10.1109/iccsnt.2013.6967182.

[4] I. Arel, D. C. Rose, and T. P. Karnowski, ''Deep machine learning—A new frontier in artificial intelligence research [research frontier],'' *IEEE Comput. Intell. Mag.*, vol. 5, no. 4, pp. 13–18, Nov. 2010, doi: 10.1109/mci.2010.938364.

[5] J. Feng, F. Li, S. Lu, J. Liu, and D. Ma, ''Injurious or noninjurious defect identification from MFL images in pipeline inspection using convolutional neural network,'' *IEEE Trans. Instrum. Meas.*, vol. 66, no. 7, pp. 1883–1892, Jul. 2017, doi: 10.1109/tim.2017.2673024.

[6] M. Khoshdeli and B. Parvin, ''Feature-based representation improves color decomposition and nuclear detection using a convolutional neural network,'' *IEEE Trans. Biomed. Eng.*, vol. 65, no. 3, pp. 625–634, Mar. 2018, doi: 10.1109/TBME.2017.2711529.

[7] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, ''Vehicle detection in satellite images by hybrid deep convolutional neural networks,'' *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 10, pp. 1797–1801, Oct. 2014, doi: 10.1109/lgrs.2014.2309695.

[8] L. Xue, X. Zhong, R. Wang, J. Yang, and M. Hu, ''Low—resolution vehicle recognition based on deep feature fusion,'' *Multimedia Tools Appl.*, vol. 77, no. 20, pp. 27617–27639, Oct. 2018.

[9] Z. Dong, M. Pei, Y. He, T. Liu, Y. Dong, and Y. Jia, ''Vehicle type classification using unsupervised convolutional neural network,'' in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, vol. 11, no. 4, pp. 172–177.

[10] 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, doi: 10.1109/tits.2015.2402438.

[11] C. M. Bautista, C. A. Dy, M. I. Manalac, R. A. Orbe, and M. Cordel, ''Convolutional neural network for vehicle detection in low resolution traffic videos,'' in *Proc. IEEE Region 10 Symp. (TENSYMP)*, May 2016, pp. 277–281, doi: 10.1109/tenconspring.2016.7519418.

[12] H. Rong and Y. Xia, ''A vehicle type recognition method based on sparse auto encoder,'' in *Proc. Int. Conf. Comput. Inf. Syst. Ind. Appl.*, 2015, pp. 323–326, doi: 10.2991/cisia-15.2015.88.

[13] G. Griffin, A. Holub, and P. Perona, ''Caltech-256 object category dataset,'' California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2007-001, 2007.

[14] Z. Huo, Y. Xia, and B. Zhang, ''Vehicle type classification and attribute prediction using multi-task RCNN,'' in *Proc. 9th Int. Congr. Image Signal Process., Biomed. Eng. Informat. (CISP-BMEI)*, Oct. 2016, pp. 564–569.

[15] S. Wang, F. Liu, Z. Gan, and Z. Cui, ''Vehicle type classification via adaptive feature clustering for traffic surveillance video,'' in *Proc. 8th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Yangzhou, China, Oct. 2016, pp. 1–5.

[16] H. Huttunen, F. S. Yancheshmeh, and K. Chen, ''Car type recognition with deep neural networks,'' in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2016, pp. 1115–1120, doi: 10.1109/ivs.2016.7535529.

[17] S. Awang, N. Mohamad, and A. Nik, ''Sparse-filtered convolutional neural networks with layer-skipping (SF-CNNLS) for intra-class variation of vehicle type recognition,'' in *Advances in Parallel Computing*, vol. 31, G. R. Joubert, Ed. Amsterdam, The Netherlands: IOS Press, 2017, pp. 194–217, doi: 10.3233/978-1-61499-822-8-194.

[18] Y.-Y. Wu and C.-M. Tsai, ''Pedestrian, bike, motorcycle, and vehicle classification via deep learning: Deep belief network and small training set,'' in *Proc. Int. Conf. Appl. Syst. Innov. (ICASI)*, May 2016, pp. 1–4, doi: 10.1109/icasi.2016.7539822.

[19] Y. Tian, W. Zhang, Q. Zhang, G. Lu, and X. Wu, ''Selective multi–convolutional region feature extraction based iterative discrimination CNN for fine–grained vehicle model recognition,'' in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 3279–3284.

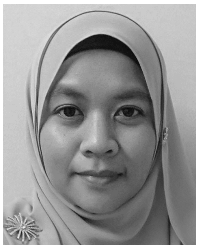
[20] M. Lin and X. Zhao, ''Application research of neural network in vehicle target recognition and classification,'' in *Proc. Int. Conf. Intell. Transp., Big Data Smart City (ICITBS)*, Jan. 2019, pp. 5–8.

[21] X. Tan, Y. Jin, G. Feng, and X. Jiang, ''Gesture segmentation based on YCrCb ellipse skin model and background subtraction,'' in *Proc. IEEE Int. Conf. Prog. Informat. Comput. (PIC)*, Dec. 2015, pp. 322–326, doi: 10.1109/pic.2015.7489862.

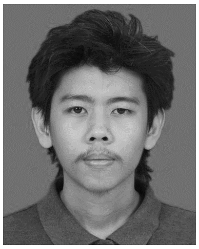
[22] R. H. M. A. Kabir, M. Rahman, M. Ahsanul, M. O. Rahman, and M. H. M. Kabir, ''A simple approach to recognize a person using hand geometry,'' *Ulab J. Sci. Eng.*, vol. 1, no. 1, pp. 19–25, 2010.

[23] Z. Chen, C. Ying, C. Lin, S. Liu, and W. Li, ''Multi–view vehicle type recognition with feedback–enhancement multi–branch CNNs,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 9, pp. 2590–2599, Sep. 2019.

[24] Y. Li, B. Song, X. Kang, X. Du, and M. Guizani, ''Vehicle–type detection based on compressed sensing and deep learning in vehicular networks,'' *Sensors*, vol. 18, no. 12, p. 4500, Dec. 2018.



SURYANTI AWANG received the Ph.D. degree in electrical engineering from Universiti Teknologi Malaysia, Johor Bahru, Malaysia, in 2014. She was a Research Officer with the Centre of Artificial Intelligence and Robotic (CAIRO), Universiti Teknologi Malaysia, from 2002 to 2005. She has been a Senior Lecturer (equivalent to Assistant Professor) with the Faculty of Computer Systems & Software Engineering, Universiti Malaysia Pahang, Malaysia, since 2005. She has coauthored for more than 30 journals and conference papers. Her research interests include pattern recognition, machine learning, and soft computing. She has collaborating with many industries in developing artificial intelligence systems. She receives numerous research grants from agencies. She has been awarded with Gold Medal in MTE'19 for Vehicle Type Recognition System, Silver Award in MTE'18, ITEX'18, and ITEX'17 for other projects.



NIK MOHAMAD AIZUDDIN NIK AZMI received the bachelor's degree in computer science from Universiti Malaysia Pahang, Pahang, Malaysia, in 2017, where he is currently pursuing the master's degree. His research interests include machine learning and bio-inspired vision.

MD. ARAFATUR RAHMAN (Senior Member, IEEE) received the Ph.D. degree in electronic and telecommunications engineering from the University of Naples Federico II, Naples, Italy, in 2013. He was a Postdoctoral Research Fellow with the University of Naples Federico II, in 2014, and a Visiting Researcher with the Sapienza University of Rome, in 2016. He is currently a Senior Lecturer (equivalent to Assistant Professor) with the Faculty of Computer Systems & Software Engineering, University Malaysia Pahang. He has coauthored of over 60 prestigious IEEE and Elsevier journal and conference publications. His research interests include the Internet-of-Things (IoT), wireless communication networks, cognitive radio networks, and vehicular communication. Dr. Rahman is a Fellow of IBM Center of Excellence, Malaysia. He has received number of prestigious international research awards, notably the Best Paper Award from ICNS'15, Italy, the Best Masters Student Award, the ITEX'17 Awards in International Exhibitions, Malaysia, and iENA'17, Germany. He has developed excellent track record of academic leadership and management and execution of international ICT projects that are supported by agencies in the Italy, EU, and Malaysia. He has served as a Publicity Chair, a Session Chair, a Programme Committee, and a member of Technical Programme Committee (TPC) in numerous leading conferences worldwide.