2014 22nd International Conference on Pattern Recognition

Vehicle Type Classification Using Unsupervised Convolutional Neural Network

Zhen Dong, Mingtao Pei, Yang He, Ting Liu, Yanmei Dong and Yunde Jia
Beijing Laboratory of Intelligent Information Technology, School of Computer Science
Beijing Institute of Technology, Beijing 100081, P.R. China
Email: {dongzhen, peimt, heyang614, liuting1989, dongyanmei, jiayunde}@bit.edu.cn

***Abstract*—In this paper, we propose an appearance-based vehicle type classification method from vehicle frontal view images. Unlike other methods using hand-crafted visual features, our method is able to automatically learn good features for vehicle type classification by using a convolutional neural network. In order to capture rich and discriminative information of vehicles, the network is pre-trained by the sparse filtering which is an unsupervised learning method. Besides, the network is with layer-skipping to ensure that final features contain both high-level global and low-level local features. After the final features are obtained, the softmax regression is used to classify vehicle types. We build a challenging vehicle dataset called BIT-Vehicle dataset to evaluate the performance of our method. Experimental results on a public dataset and our own dataset demonstrate that our method is quite effective in classifying vehicle types.**

***Keywords*—vehicle type classification; convolutional neural network; sparse filtering;**

I. Introduction

Vision-based vehicle type classification is one of the essential tasks in Intelligent Traffic System (ITS), and has a multitude of applications, such as traffic volume and speed estimation, illegal vehicle type detection, and incident detection. Numerous methods of vehicle type classification have been proposed, and most of them fall into two categories: model-based methods and appearance-based methods. Model-based methods [1][2][3][4] recover the vehicle's 3D parameters such as length, width, and height for classification from multi-view images. Appearance-based methods extract appearance features (e.g. SIFT [5], Sobel edges [6]) from either vehicle frontal or side view image to classify vehicle types. In real applications, more and more vehicle frontal view images are captured by traffic surveillance cameras in many places such as checkpoints and intersections. In this paper, we propose an appearance-based vehicle type classification method from vehicle frontal view images.

Compared to the appearance-based methods using vehicle side view images [7][8][9][10], only few work has been reported on vehicle type classification from vehicle frontal view images. Petrovic *et al.* [11] extracted a great many of features, such as Sobel edge response, edge orientation, direct normalized gradients, locally normalized gradients, and Harris corner response from vehicle frontal view images to classify vehicle types. Negri *et al.* [12] presented a voting algorithm for a multiclass vehicle type recognition system based on oriented-contour points. Psyllos *et al.* [13] used SIFT features to recognize the logo, manufacture and model of a vehicle. Peng *et al.* [14] represented a vehicle by license plate color, vehicle front width, and type probabilities for vehicle type classification. However, these methods use multiple hand-crafted features and face the problem that the visual features extracted are not discriminative enough. To solve this problem, we propose to learn good features for vehicle type classification.

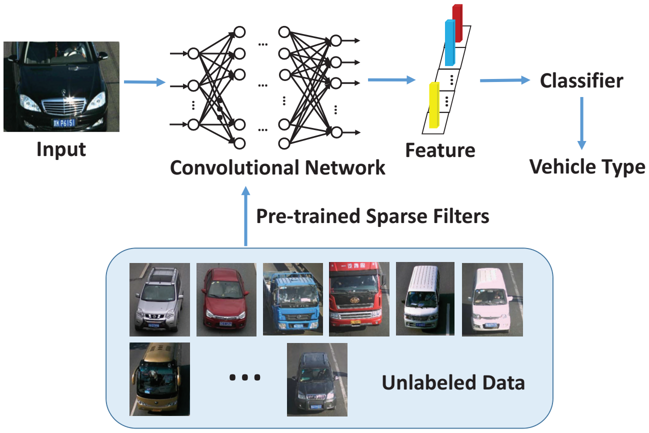


Fig. 1. The framework of our method. For each image with a vehicle, the convolutional neural network which is pre-trained by the sparse filtering [15] is first utilized to learn features. The features are then used to classify vehicle types by the softmax regression.

Convolutional neural networks (ConvNets) [16] are multi-layer feed-forward neural networks whose structures are biologically-inspired. Unlike other vision methods using hand-crafted features, ConvNets are able to automatically learn multiple stages of invariant features for the specific task. ConvNets thus have been utilized in a great deal of applications such as hand tracking [17], face detection [18][19], facial point detection [20], and image classification [21].

As shown in Fig.1, we use a convolutional neural network to learn good features for vehicle type classification. As unsupervised pre-training has shown great usefulness in training multi-layer neural networks [22][23][24][25], we also provide an unsupervised pre-training procedure for the network. The sparse filtering [15] is introduced to pre-train a great number of filters. As essential parts of the convolutional neural network, these filters are able to capture rich and discriminative information of vehicles for improving classification performances. The convolutional neural network in our method contains two stages which generate low-level local and high-level global features respectively. Unlike traditional ConvNets, our network is with layer-skipping (*i.e.*, the outputs of the two stages

1051-4651/14 $31.00 © 2014 IEEE
DOI 10.1109/ICPR.2014.39

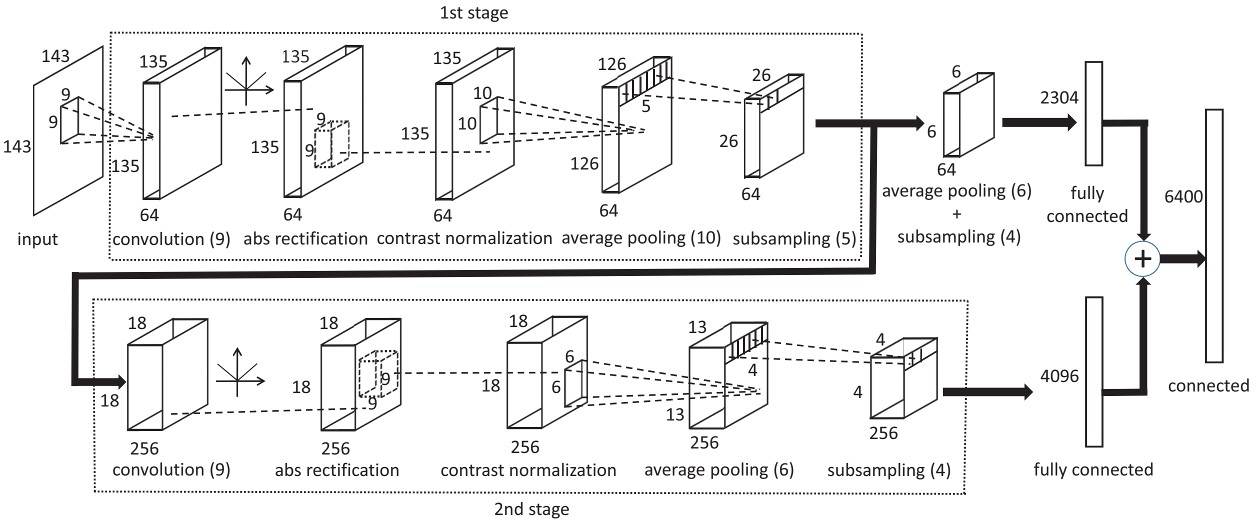


Fig. 2. The architecture of our unsupervised convolutional neural network. The network contains two stages, each of which is consisted of convolutional, absolute rectification, local contrast normalization, average pooling, and subsampling layers. The number behind ”convolution” is the size of the unsupervised pre-trained sparse filters. The number behind ”average pooling” is the size of the average filter, and the number behind ”subsampling” reflects the step of subsampling.

are all fed into the classifier), which allows the classifier use high-level global as well as low-level local features. The high-level global features provide holistic descriptions of the vehicle, and the low-level features aim to characterize vehicle parts precisely. Both of them are beneficial for vehicle type classification. After the final features are obtained, the softmax regression is used to classify vehicle types.

II. Feature Learning

The architecture of our convolutional neural network is shown in Fig.2. The network contains two stages, each of which is consisted of convolutional, absolute value rectification, local contrast normalization, average pooling, and subsampling layers.

A. Convolutional Layer

The convolutional layer computes convolutions of the input with a series of filters. As a result, it is also called the filter bank layer. The input of the convolutional layer is a 3D array whose size is $s1 \times s2 \times s3$, where s1 is the number of 2D features, and $s2 \times s3$ is the size of the 2D feature. The output is also a 3D array with the size of $t1 \times t2 \times t3$. Suppose the filter size is $l1 \times l2$, we thus have $t2 = s2 - l1 + 1$ and $t3 = s3 - l2 + 1$ due to the board effects.

Here, we introduce the Sparse Filtering [15] to learn the filter bank. Sparse filtering is an unsupervised feature learning algorithm which is easy to implement and hyperparameter-free. It optimizes for the sparsity in the feature distribution and avoids explicit modeling of the data distribution. This gives rise to a simple formulation and permits the effectiveness of learning. Additionally, the sparse filtering only has one hyperparameter, the number of features to learn, which can be easily tuned.

Define a data matrix as $\mathcal{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n] \in \mathbb{R}^{d\times n}$, where each column is a data point. Similarly, $\mathcal{F} \in \mathbb{R}^{t\times n}$ is a feature distribution matrix over $\mathcal{X}$, where each row is a feature and each column is an example. The element $\mathcal{F}_{ij}$ represents the activity of the i-th feature on the j-th example of $\mathcal{F}$. By endowing $\mathcal{F}$ with sparse constrains, our goal is to obtain a $\mathcal{S} = [\boldsymbol{s}_1, \boldsymbol{s}_2, \cdots, \boldsymbol{s}_t] \in \mathbb{R}^{d\times t}$ which satisfies $\mathcal{F} = \mathcal{S}^\top \mathcal{X}$. We thus have $\mathcal{F}_{ij} = \boldsymbol{s}_i^T \boldsymbol{x}_j$, so each column of $\mathcal{S}$ can be viewed as a filter.

Let $\boldsymbol{f}_{(i,\triangle)} \in \mathbb{R}^{1\times n}(i = 1, 2, \cdots, t)$ represents the i-th row of $\mathcal{F}$, and $\boldsymbol{f}_{(\triangle,j)} \in \mathbb{R}^{t\times 1}(j = 1, 2, \cdots, n)$ represents the j-th column of $\mathcal{F}$. The objective function of the sparse filtering method is computed in three steps: normalizing the feature distribution matrix by rows, normalizing the feature distribution matrix by columns, and summing up the absolute values of all entries. Specifically, in the first step, each feature is divided by its $\mathcal{L}2$-norm across all examples: $\widetilde{\boldsymbol{f}}_{(i,\triangle)} = \boldsymbol{f}_{(i,\triangle)}/\|\boldsymbol{f}_{(i,\triangle)}\|_2$. In this way, each feature is normalized to be equally active. In the second step, each example is divided by its $\mathcal{L}2$-norm across all features: $\hat{\boldsymbol{f}}_{(\triangle,j)} = \widetilde{\boldsymbol{f}}_{(\triangle,j)}/\|\widetilde{\boldsymbol{f}}_{(\triangle,j)}\|_2$ to ensure all examples lie on the unit $\mathcal{L}2$-ball. In the third step, we sum up the absolute values of all the entries in $\hat{\mathcal{F}}$. The sparse filtering is described as

$$\min_{\mathcal{S}} \|\hat{\mathcal{F}}\| = \sum_{i=1}^{t}\sum_{j=1}^{n} |\hat{\mathcal{F}}_{ij}|. \qquad (1)$$

Eq.(1) equals to

$$\min_{\mathcal{S}} \sum_{j=1}^{n} \|\hat{\boldsymbol{f}}_{(\triangle,j)}\|_1 = \sum_{j=1}^{n} \left\| \frac{\widetilde{\boldsymbol{f}}_{(\triangle,j)}}{\|\widetilde{\boldsymbol{f}}_{(\triangle,j)}\|_2} \right\|_1. \qquad (2)$$

The sparse objective makes the feature distribution have three properties, which have been explored in the neuroscience literatures [26][27].

Population Sparsity: It means that each example should only have a few active (non-zero) features. This property called the population sparsity [26][27] is considered to be an efficient means of coding in early vision cortex. The term $\|\hat{\boldsymbol{f}}_{(\triangle,j)}\|_1$ in Eq.(2) reflects the population sparsity property of the features on the j-th example. As $\hat{\boldsymbol{f}}_{(\triangle,j)}$ has been constrained to lie on the unit $\mathcal{L}2$-ball, the objective function is minimized when the features are sparse. In other words, the objective tends to place the examples close to feature axes, and the example which has similar values on each feature would have a high penalty.

High Dispersal: The feature distribution should have similar statistics on different features. Here, the statistics are taken as the mean squared activations by averaging the squared values in the feature matrix across all the examples:

$$\mathcal{T}_i = \frac{1}{n}\sum_{j=1}^{n} \mathcal{F}_{ij}^2 = \frac{1}{n}\|\boldsymbol{f}_{(i,\triangle)}\|_2^2. \tag{3}$$

The statistics for different features should be roughly same which implies that the contributions of all features are roughly same. This principle is known as the high dispersal. In the first step of computing the objective function, each feature is divided by its $\mathcal{L}2$-norm across all examples which ensures that each feature is normalized to be equally active. Therefore, the objective optimizes for high dispersal.

Lifetime Sparsity: The property that each feature should be active only on a few examples is called lifetime sparsity [26][27]. This property guarantees that the feature is discriminative enough to distinguish different examples. Specifically, each row of the feature distribution matrix should only have a few active (non-zero) entries. In the sparse filtering algorithm, the lifetime sparsity property is ensured by the population sparsity and high dispersal properties. The feature distribution matrix should have a great many non-active (zero) elements due to the population sparsity property. These non-active elements could not be placed in a few specific rows, otherwise it would be against the high dispersal property. Therefore, each feature would have a significant number of non-active elements and thus be lifetime sparse.

The sparse filtering is easy to implement. The minimization of Eq.(2) is achieved by L-BFGS optimization method. As the objective function contains the absolute value operators which are nondifferentiable, an approximation is introduced. The absolute value operators are ignored, and the gradient of the objective function is computed by the back-propagation algorithm.

For each stage of the convolutional neural network, the filter size is set to be 9×9 pixels. Many patches size of 9×9 are randomly extracted from the input, and the sparse filtering method is utilized to learn the filters. The learnt filters of the 1-st stage are shown in Fig.3. As we expected, the filters mainly preserve the edge, point and junction information of the vehicles. A few filters don't achieve convergency due to the computation cost.

B. Absolute Value Rectification Layer

In this module, all the elements are passed into the absolute value rectification function:

$$\boldsymbol{y}_{ijk} = |\boldsymbol{x}_{ijk}|. \tag{4}$$

This is inspired by the fact that the relationship between two items in real world is always positive or zero, but not negative.

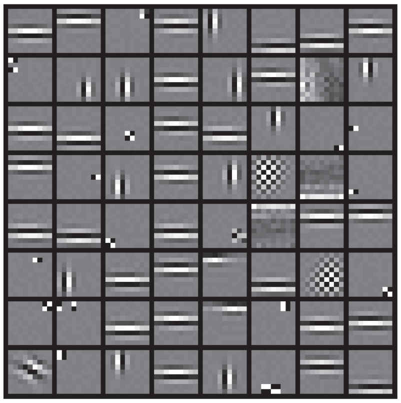


Fig. 3. The unsupervised pre-trained sparse filters of the 1-st stage.

Several rectifying strategies including the absolute value and the positive part are tried in [24], and the results are similar. We choose the absolute value function here.

C. Local Contrast Normalization Layer

The local contrast normalization layer is inspired by computational neuroscience [28][29]. The goal of this layer is to enforce local competitions between one neuron and its neighbors, including the neurons nearby in the same feature map and the ones at the same 2D location in different feature maps. To achieve this goal, two normalization operations are performed: subtractive and divisive. For the element $x_{i,j,k}$ in the input 3D array size of $I \times J \times K$, the subtractive normalization operator computes

$$y_{i,j,k} = x_{i,j,k} - \sum_{o,p,q} \omega_{p,q} x_{o,j+p,k+q}, \tag{5}$$

where $\omega_{p,q}$ is a normalized Gaussian filter with the size of 9×9, so $\sum_{p,q} \omega_{p,q} = 1$. In addition, the extra zeros are need to add to the boards of the input while filtering. The divisive normalization operator is performed as

$$z_{i,j,k} = \frac{y_{i,j,k}}{max\Big(M, \sqrt{\sum_{o,p,q} \omega_{p,q} y_{o,j+p,k+q}^2}\Big)}, \tag{6}$$

where

$$M = \frac{1}{J \times K}\sum_{j=1}^{J}\sum_{k=1}^{K}\sqrt{\sum_{o,p,q} \omega_{p,q} y_{o,j+p,k+q}^2}. \tag{7}$$

D. Average Pooling and Subsampling Layers

The average pooling and subsampling layers aim to make the representation robust to geometric distortions and small shifts, similar to the ”complex cells” in standard models of the visual cortex. In the average pooling layer, we compute the convolutions of each 2D feature map with an average filter

$$y_{i,j,k} = \sum_{p,q} \alpha_{p,q} x_{i,j+p,k+q}, \tag{8}$$

where $\alpha_{p,q} = 1/(f1 \times f2)$ is the average filter with the size of $f1 \times f2$. The subsampling procedure is then performed with the steps of $p1$ horizontally and $p2$ vertically as shown in Fig.2. Therefore, suppose the input 3D array of these two layers are size of $m \times s1 \times s2$, the output is also a 3D array with the size of $m \times t1 \times t2$ and

$$\begin{aligned} t1 &= \left\lfloor \frac{s1 - f1}{p1} \right\rfloor + 1, \\ t2 &= \left\lfloor \frac{s2 - f2}{p2} \right\rfloor + 1. \end{aligned} \tag{9}$$

III. Classification

We use the softmax regression to classify vehicle types by features learnt above. The relationship between the feature and the probability distribution of its category is modeled by

$$\boldsymbol{v} = W^\top \boldsymbol{x} + \boldsymbol{b}, \tag{10}$$

where $\boldsymbol{x} \in \mathbb{R}^{D\times 1}$ represents the input feature, $\boldsymbol{v} \in \mathbb{R}^{C\times 1}$ is computed to describe the distribution, and C is the number of categories. Because the probability has the properties of nonnegativity and unitarity, the following normalization is performed:

$$\begin{aligned} \boldsymbol{d}_i &= \frac{1}{V} e^{\boldsymbol{v}_i}, i = 1, 2, \cdots, C, \\ V &= \sum_{i=1}^{C} e^{\boldsymbol{v}_i}, \end{aligned} \tag{11}$$

where $\boldsymbol{v}_i$ is the i-th element of $\boldsymbol{v}$, and $\boldsymbol{d} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \cdots, \boldsymbol{d}_C]^\top$ is the output of the model.

Denote the training samples as $\{(\boldsymbol{x}^{(i)}, \boldsymbol{y}^{(i)}|i = 1, 2, \cdots, N\}$ where $\boldsymbol{x}^{(i)}$ is the learnt feature, $\boldsymbol{y}^{(i)}$ is the probability distribution of $\boldsymbol{x}^{(i)}$'s category. Specifically, if $\boldsymbol{x}^{(i)}$ belongs to the j-th class $(1 \le j \le C)$, the j-th element of $\boldsymbol{y}^{(i)}$ will be 1 and others will be 0. We want to learn the parameters, W and $\boldsymbol{b}$, in the model by using these training samples. Two principles are introduced to learn the parameters, the first one is the Kullback-Leibler divergence (KL):

$$\begin{aligned} &\min_{W,\boldsymbol{b}} \sum_{i=1}^{N} KL(\boldsymbol{y}^{(i)} \| \boldsymbol{d}^{(i)}) \\ &= \min_{W,\boldsymbol{b}} \sum_{i=1}^{N} \left(\sum_{j=1}^{C} \boldsymbol{y}_j^{(i)} \ln \frac{1}{\boldsymbol{d}_j^{(i)}} - \sum_{j=1}^{C} \boldsymbol{y}_j^{(i)} \ln \frac{1}{\boldsymbol{y}_j^{(i)}} \right) \\ &= \min_{W,\boldsymbol{b}} - \sum_{i=1}^{N} \sum_{j=1}^{C} \boldsymbol{y}_j^{(i)} \ln \boldsymbol{d}_j^{(i)}, \end{aligned} \tag{12}$$

and the second one is the Minimum Squared Error (MSE):

$$\min_{W,\boldsymbol{b}} \sum_{i=1}^{N} \frac{1}{2} \| \boldsymbol{d}^{(i)} - \boldsymbol{y}^{(i)} \|_2^2. \tag{13}$$

The online gradient descent is utilized to solve Eq.(12) and Eq.(13). After the optimized parameters are obtained, the class of a test sample can be predicted by picking the label which makes the probability achieving maximum.

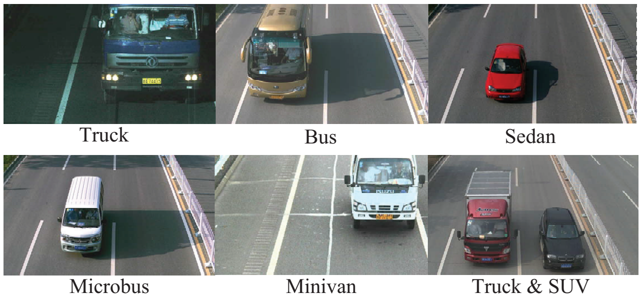


Fig. 4. The example images of BIT-Vehicle Dataset. All vehicles in our dataset fall into 6 types: Bus, Microbus, Minivan, Sedan, SUV, and Truck.

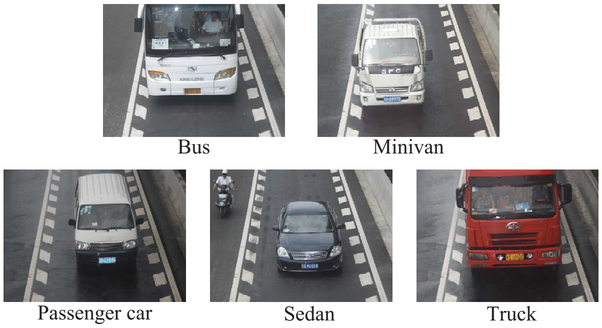


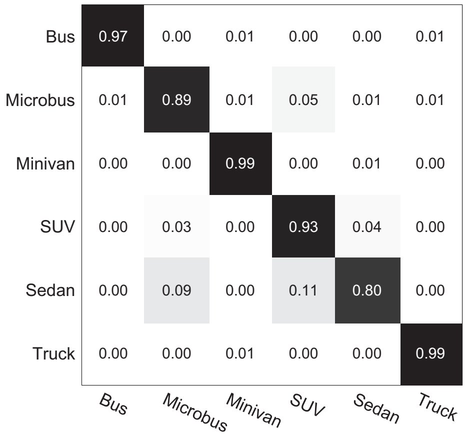
Fig. 5. The example images of the dataset in [30]. This dataset consists of 5 types of vehicles: Bus, Minivan, Passenger car, Sedan, and Truck.

IV. Experiments

A. Datasets and Preprocessing

We collected a complex and challenging vehicle dataset called BIT-Vehicle Dataset to test our method. Fig.4 shows the example images of our dataset. As shown in the figure, images in this dataset are captured at different places and different time. These images contain changes in illumination condition, scale, the surface color of vehicles and viewpoint. The top or bottom parts of some vehicles are not included in the images because of the capturing delay and the size of the vehicle. In addition, these images have two kinds of sizes: 1600×1200 and 1920×1080. All vehicles in the dataset are divided into six categories: Bus, Microbus, Minivan, Sedan, SUV, and Truck. Each category contains 150 vehicles, to provide a total of 900 vehicles. In order to give a better estimation of the generalization performance, we use the 10-fold cross-validation scheme in the experiment. Additionally, the vehicle whose location is pre-annotated is regarded as the input of the convolutional neural network.

To make a fair comparison with the previous methods, we also test our method on the dataset released in [30]. There are totally 3618 daylight and 1306 nightlight images with the image size of 1600×1264 in this dataset. All the images are captured in highways with a fixed camera. The vehicles in images fall into five categories: Truck, Minivan, Bus, Passenger car, and Sedan(including sport-utility vehicle (SUV)). Example images are displayed in Fig.5. As shown in the figure, this dataset contains some challenging factors, including illumination variations, rain blurring, different color surfaces



	Bus	Microbus	Minivan	SUV	Sedan	Truck
Bus	0.97	0.00	0.01	0.00	0.00	0.01
Microbus	0.01	0.89	0.01	0.05	0.01	0.01
Minivan	0.00	0.00	0.99	0.00	0.01	0.00
SUV	0.00	0.03	0.00	0.93	0.04	0.00
Sedan	0.00	0.09	0.00	0.11	0.80	0.00
Truck	0.00	0.00	0.01	0.00	0.00	0.99

Fig. 6. The confusion matrix of our method on BIT-Vehicle Dataset.

of vehicles, and background interferences. The reported results of this dataset are the averages of 10 independent experiments.

For both datasets, a preprocess procedure similar to [29] is performed to the images. First, all images are converted to gray and resized to a new image with the longer side size of 151 pixels and the height-width ratio fixed. The mean of the image is then subtracted from each pixel and the standard deviation is divided. In the third step, the local contrast normalization is performed similar to Sec.II-C except $M = 1$ and that no extra zeros are added when computing Eq.(5). The length of the longer side is thus to be $151 - 9 + 1 = 143$ pixels. Finally, the zero-padding is executed for the shorter side to 143 pixels to ensure that the input of the network is square.

B. Results on BIT-Vehicle Dataset

In this part, we test our method on BIT-Vehicle dataset and report its performance. Our approach achieves 92.89% accuracy. As we expected, our model can precisely classify vehicle types in some challenging situations, such as different lighting conditions, serious shadow interferences, and viewpoint changes. The primary reason is that our convolutional neural network is able to learn discriminative features for vehicle type classification. The confusion matrix is shown in Fig.6. From the matrix, we find that most of the misclassifications are among ”Microbus”, ”SUV” and ”Sedan”. This is because they have quite similar appearances. Additionally, the fact that the classifier we use is quite simple demonstrates that the learnt features are discriminative and reliable to classify vehicle types.

We also evaluate the effects of several components involved in the convolutional neural network. At first, the unsupervised pre-trained sparse filters are replaced by random values. The classification accuracy is displayed in TABLE I. It shows that the network with unsupervised pre-trained sparse filters outperforms that with random filters. The performance gain achieved by our model due to the sparse filters learnt from unlabeled vehicles are able to capture rich discriminative information of vehicles for vehicle type classification.

We further investigate the contribution of the network depth. Here, we compare the performances of two types of network architecture. For the first one, only the 1-st stage is utilized to learn vehicle features, and the dimensionality of the final feature is thus 2304. For the second one, the 2-nd stage is added but without layer-skipping strategy, and the dimensionality of the final feature is thus 4096. Their average classification accuracies are also shown in TABLE I. The performance difference between them emphasizes that multi-stage is better than one stage for vehicle feature learning.

TABLE I. CLASSIFICATION ACCURACIES ON BIT-VEHICLE DATASET.

Method	Accuracy(%)	
	KL	MSE
Random Filter	87.56	88.22
Only 1st stage	89.78	90.22
1st stage+2nd stage without layer-skipping	90.22	91.33
Our model	92.67	92.89

At last, the benefit of the layer-skipping strategy is verified. The layer-skipping strategy connects the features learnt from the 1-st stage and the 2-nd stage. The features learnt from the 1-st stage are low-level and local, and the outputs of the 2-nd stage are high-level global features. From TABLE I, we find that our model outperforms the model only using 1-st stage. Our model uses the high-level global features to capture rich and discriminative information, while the model only with one stage focuses on local parts of vehicles and lacks the holistic description of the vehicle. Similarly, the contribution of low-level local features can be seen from the performance difference between our model and the model without layer-skipping. With low-level local features, our model is able to describe the details of the vehicle precisely. Therefore, these two types of features can be effectively utilized by adding the layer-skipping strategy into the network.

C. Comparison Results

To compare with previous methods on vehicle type classification, we test our method on the dataset released in [30]. For fair consideration, the experiments on daylight images and nightlight images are performed respectively, which is similar to [30]. As there is only one vehicle in each image of this dataset, the image is immediately thrown into our convolutional neural network to learn features without vehicle detection. The comparison results are shown in TABLE II where the results of other methods are available from published papers conveniently. The comparison results show that our method achieves 95.7% classification accuracy on daylight images and 88.8% on nightlight images, better than the results of previous methods. The underlying reason is that the convolutional neural network we use is able to learn discriminative and reliable features for vehicle type classification. The unsupervised pre-trained sparse filters can capture rich and discriminative information of vehicles. In addition, the layer-skipping strategy allows the classifier use both high-level global and low-level local features. Their advantages can be effectively utilized in classifying vehicle types. It should be noted that our method even without vehicle detection outperforms other methods. Therefore, our method is effective in classifying vehicle types.

TABLE II. COMPARISON RESULTS ON THE DATASET IN [30].

Method	Accuracy(%)	
	Daylight	Nightlight
Psyllos *et al.* [13]	78.3	73.3
Petrovic *et al.* [11]	84.3	82.7
Peng *et al.* [30]	90.0	87.6
Dong *et al.* [31]	91.3	–
Peng *et al.* [14]	93.7	–
Ours	95.7	88.8

V. CONCLUSION

We have proposed an appearance-based vehicle type classification method from vehicle frontal view images by using an unsupervised convolutional neural network. Our model which is pre-trained by the sparse filtering can automatically learn discriminative and reliable features for vehicle type classification. The layer-skipping strategy in the network ensures that the final features are consisted of both high-level global and low-level local features. After the final features are obtained, the softmax regression is utilized to classify vehicle types. Experimental results on two datasets showed the effectiveness of our method on vehicle type classification.

ACKNOWLEDGMENT

This work was supported in part by the Natural Science Foundation of China (NSFC) under grant No. 61203291, the Specialized Research Fund for the Doctoral Program of Higher Education of China (20121101120029), and the Specialized Fund for Joint Building Program of Beijing Municipal Education Commission.

REFERENCES

[1] A. H. Lai, G. S. Fung, and N. H. Yung, "Vehicle type classification from visual-based dimension estimation," in *Intelligent Transportation Systems Conferecnce Proceedings*. IEEE, 2001, pp. 201–206.

[2] S. Gupte, O. Masoud, R. F. Martin, and N. P. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37–47, 2002.

[3] J.-W. Hsieh, S.-H. Yu, Y.-S. Chen, and W.-F. Hu, "Automatic traffic surveillance system for vehicle tracking and classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 2, pp. 175–187, 2006.

[4] Z. Zhang, T. Tan, K. Huang, and Y. Wang, "Three-dimensional deformable-model-based localization and recognition of road vehicles," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 1–13, 2012.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, January 2004.

[6] I. Sobel, "Camera models and machine perception," DTIC Document, Tech. Rep., 1970.

[7] X. Ma and W. E. L. Grimson, "Edge-based rich representation for vehicle classification," in *International Conference on Computer Vision*, vol. 2. IEEE, 2005, pp. 1185–1192.

[8] C. Zhang, X. Chen, and W.-b. Chen, "A pca-based vehicle classification framework," in *22nd International Conference on Data Engineering Workshops*. IEEE, 2006, pp. 17–17.

[9] P. Ji, L. Jin, and X. Li, "Vision-based vehicle type classification using partial gabor filter bank," in *International Conference on Automation and Logistics*. IEEE, 2007, pp. 1037–1040.

[10] Y. Shan, H. S. Sawhney, and R. Kumar, "Unsupervised learning of discriminative edge measures for vehicle matching between nonoverlapping cameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 4, pp. 700–711, 2008.

[11] V. S. Petrovic and T. F. Cootes, "Analysis of features for rigid structure vehicle type recognition." in *British Machine Vision Conference*, 2004, pp. 1–10.

[12] P. Negri, X. Clady, M. Milgram, and R. Poulenard, "An oriented-contour point based voting algorithm for vehicle type classification," in *International Conference on Pattern Recognition*, vol. 1. IEEE, 2006, pp. 574–577.

[13] A. Psyllos, C.-N. Anagnostopoulos, and E. Kayafas, "Vehicle model recognition from frontal view image measurements," *Computer Standards & Interfaces*, vol. 33, no. 2, pp. 142–151, June 2011.

[14] Y. Peng, J. S. Jin, S. Luo, M. Xu, S. Au, Z. Zhang, and Y. Cui, "Vehicle type classification using data mining techniques," in *The Era of Interactive Media*. Springer, 2013, pp. 325–335.

[15] J. Ngiam, Z. Chen, S. A. Bhaskar, P. W. Koh, and A. Ng, "Sparse filtering," in *Advances in Neural Information Processing Systems*, 2011, pp. 1125–1133.

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

[17] S. J. Nowlan and J. C. Platt, "A convolutional neural network hand tracker," *Advances in Neural Information Processing Systems*, pp. 901–908, 1995.

[18] C. Garcia and M. Delakis, "A neural architecture for fast and robust face detection," in *16th International Conference on Pattern Recognition*, vol. 2. IEEE, 2002, pp. 44–47.

[19] M. Osadchy, Y. L. Cun, and M. L. Miller, "Synergistic face detection and pose estimation with energy-based models," *The Journal of Machine Learning Research*, vol. 8, pp. 1197–1215, 2007.

[20] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 3476–3483.

[21] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1106–1114.

[22] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[23] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," *Advances in neural information processing systems*, vol. 19, p. 153, 2007.

[24] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *International Conference on Computer Vision*. IEEE, 2009, pp. 2146–2153.

[25] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. L. Cun, "Learning convolutional feature hierarchies for visual recognition," in *Advances in neural information processing systems*, 2010, pp. 1090–1098.

[26] D. J. Field, "What is the goal of sensory coding?" *Neural computation*, vol. 6, no. 4, pp. 559–601, 1994.

[27] B. Willmore and D. J. Tolhurst, "Characterizing the sparseness of neural codes," *Network: Computation in Neural Systems*, vol. 12, no. 3, pp. 255–270, 2001.

[28] S. Lyu and E. P. Simoncelli, "Nonlinear image representation using divisive normalization," in *Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[29] N. Pinto, D. D. Cox, and J. J. DiCarlo, "Why is real-world visual object recognition hard?" *PLoS computational biology*, vol. 4, no. 1, p. e27, 2008.

[30] Y. Peng, J. S. Jin, S. Luo, M. Xu, and Y. Cui, "Vehicle type classification using pca with self-clustering," in *International Conference on Multimedia and Expo Workshops*. IEEE, 2012, pp. 384–389.

[31] D. Zhen and Y. Jia, "Vehicle type classification using distributions of structural and appearance-based features," in *International Conference on Image Processing*. IEEE, 2013, pp. 4321–4324.