Fine-Grained Vehicle Model Recognition Using A Coarse-to-Fine Convolutional Neural Network Architecture

Jie Fang, *Student Member, IEEE*, Yu Zhou, *Member, IEEE*, Yao Yu, *Member, IEEE*, and Sidan Du, *Member, IEEE*

***Abstract*— Fine-grained vehicle model recognition is a challenging problem in intelligent transportation systems due to the subtle intra-category appearance variation. In this paper, we demonstrate that this problem can be addressed by locating discriminative parts, where the most significant appearance variation appears, based on the large-scale training set. We also propose a corresponding coarse-to-fine method to achieve this, in which these discriminative regions are detected automatically based on feature maps extracted by convolutional neural network. A mapping from feature maps to the input image is established to locate the regions, and these regions are repeatedly refined until there are no more qualified ones. The global and local features are then extracted from the whole vehicle images and the detected regions, respectively. Based upon the holistic cues and the subordinate-level variation within these global and local features, an one-versus-all support vector machine classifier is applied for classification. The experimental results show that our framework outperforms most of the state-of-the-art approaches, achieving 98.29% accuracy over 281 vehicle makes and models.**

***Index Terms*— Fine-grained vehicle recognition, convolutional neural network, one-versus-all SVM.**

I. Introduction

BEYOND identifying general category, fine-grained vehicle model recognition, i.e., identifying manufacture and detailed model of a vehicle, plays a more versatile role in ITS. For example, vehicle model attributes can be used as key words to search target vehicle in traffic surveillance images, even when the number plate of the vehicle is faked. Electronic toll collection can also charge vehicles automatically through recognizing their different models. Vehicle models can be used to analyse and study traffic flow in order to make strategies of traffic control. Despite the importance of fine-grained vehicle model recognition, most traditional vehicle recognition systems focus only on Vehicle Make Recognition(VMR) and recognizing general categories such as sedan, minivan, SUV, truck and so on. Numerous methods have been proposed in such fields, these approaches primarily take advantage of 3D Model [1]–[3] and low-level features [4]–[9], such as SIFT, HOG and SPM to identify the vehicle makes or categories.

Different than these traditional topics, vision-based fine-grained vehicle model recognition is a tough task due to subtle appearance variation among subordinate-level vehicle categories, with which the fine-grained model cannot be easily recognized even by a human without domain knowledge. Thus, fine-grained vehicle model recognition needs more powerful and discriminative features to classify intra-class objects. This imposes a great challenge in this domain, leading to few works of fine-grained vehicle model recognition in ITS. Prokaj and Medioni [10] took a model-based, top-down approach to classify vehicles, which utilizes 3-D models to match the features in the model images. Ramnath *et al*. [11] presented a new approach for recognizing the vehicel types from a single image using 3D curve alignment, which makes their system can be able to verify a type from an arbitrary view. Zhan *et al*. [12] extracted Harr-like features from the training samples to build global appearance model for fine-grained feature representation, and then train the data by a multi-class SVMs classifier to distinguish the intra-class variation of cars. In Zhang's work [13], a highly reliable classification scheme was proposed as a cascade ensemble classifier with reject option to accommodate the situations, when no decision should be made once there exists adequate ambiguity. Clady *et al.* [14] presented a framework for multiclass vehicle type identification based on oriented contour points. In this framework, three voting algorithms and a distance error allowed to measure the similarity between an input instance and the databases classes, which can be combined to design a discriminant function. In some of these approaches, the experimental databases consist of only a few subordinate-level categories resulting in a tight limitation of application. Moreover, most of these works heavily rely on hand-crafted low-level features which might not be saliently distinctive among different subordinate-level categories that have extremely similar appearance.

Recently, some researchers realize the fact that features extracted from a few important parts may infer more discriminative information than the ones from the whole vehicle image. Lee [15] proposed an approach which exploits texture descriptors computed from the front images of vehicles.

Manuscript received January 26, 2016; revised May 25, 2016 and October 5, 2016; accepted October 18, 2016. Date of publication November 15, 2016; date of current version June 26, 2017. This work was supported in part by the Natural Science Foundation of Jiangsu Province under Grant BE2015152 and in part by the National Natural Science Foundation of China, under Grant 61100111, Grant 61300157, Grant 61201425, and Grant 61271231. The Associate Editor for this paper was D. Fernandez-Llorca.

The authors are with the School of Electronic Science and Engineering, Nanjing University, Nanjing 210046, China (e-mail: mg1523017@smail.nju.edu.cn; nackzhou@nju.edu.cn; allanyu@nju.edu.cn; coff128@nju.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2016.2620495

Psyllos *et al*. [16] used a probabilistic Neural Network for recognition using vehicle frontal view images. He *et al*. [17] proposed a recognition framework of vehicle manufactures and models, where front parts are detected to extract features for recognition. Llorca *et al*. [18] provided a novel approach by learning the geometry and the appearance of car emblems from rear view images. Siddiqui *et al*. [19] proposed an approach based on the Bag-of-Features paradigm for representing vehicles' front/rear parts. Yang *et al.* [20] applied CNN to different viewpoints of each vehicle respectively for fine-grained vehicle model recognition. Despite of their outperforming results in their experiments, most of these methods highlight the important parts of vehicle by utilizing human's empirical sense, therefore, it may fail in generalizing to others beyond the experimental databases. Moreover, from the results of our research, recognition may benefits more from some well-located detail parts than the front or rear parts, which however are not well studied by all the aforementioned researches.

Inspired by this finding, in this paper, we propose a coarse-to-fine CNN for fine-grained vehicle model recognition, in which the most discriminant parts are automatically detected via feature maps generated by CNN. Other than the manual design, our method can learn which parts of image are important for identifying subordinate-level model variation. Besides, in contrast to hand-crafted features used in previous methods, CNN is able to learn hierarchical features from large-scale dataset using its multi-layer feed-forward structure. Our feature learning process is coarse-to-fine, detecting more and more subtle parts, and extracting features from both the global and local regions respectively. Eventually, the global features can imply holistic cues, while the local features can describe subordinate-level variation. Based upon the learned features, an one-versus-all SVM classifier is applied for classification. The experimental results show that our framework outperform most of the state-of-the-art approaches, achieving 98.29% accuracy over 281 vehicle makes and models. The contribution of our work are summarized as follows:

- We propose a coarse-to-fine CNN framework for fine-grained vehicle model recognition, which can hierarchically extract the discriminative features of both the global and local regions and achieve the state-of-the-art results.
- We propose a novel method for detecting the most discriminative parts, which makes our system more flexible and generic so that can also be used in other similar recognition tasks to improve system performance and reduce the cost of feature design.

The rest of this paper is organised as follows. In Section II, we detail the architecture of our coarse-to-fine CNN. The experimental results and analysis are demonstrated in Section III. And finally, we conclude our work in Section IV.

II. Framework of Proposed Method

The framework of our fine-grained vehicle model recognition is illustrated in Fig. 1. It's a coarse-to-fine system where part sets $P_1, P_2, \ldots, P_n$ are detected automatically by *parts detection*, which is our critical contribution and represented more concretely in Fig. 2. *Part detection* is achieved by *CNN training* and *parts locating*. In our experiments, we propose a CNN similar to the architecture in [21], which contains five convolution layers, three pooling layers, three local response normalization layers and three fully connected layers. The CNN is illustrated in Fig. 2. After training the network, an average feature map F_i of the entire input set is generated. This feature map is used to detect Regions of Interest(RoI) and these RoI are acquired by our *parts locating*. The results of the detecting procedure are further regarded as the input set to obtain more subtle regions, and such procedure will keep running until there are no more regions to obtain. These critical parts can be learned automatically with no any human's consciousness required. This advantage makes our system more intelligent and applicable for other fine-grained recognition tasks. Then, both the global and local features are combined to train an one-versus-all linear SVM.

A. Convolutional Neural Network

CNN is an end-to-end system, in which the input is a raw image, while the output is a prediction through the distinctive features extracted via intermediate layers. Compared with many traditional methods depend on hand-engineered features or separately trained by machine learning algorithms, CNN exhibits its powerful ability of extracting features automatically and optimizing the whole system conveniently.

1) Convolution Layers: Convolution layers are used to convolve previous layer's feature maps with multiple filter masks to extract features and feed the activation function to generate the output feature maps, which are also the input of the next layer. There are two important concepts, local connectivity and parameter sharing. Local connectivity means each filter will convolve only a local region of the input volume in order to decrease the number of the weight parameters. It is inspired by biological systems. The spatial extent of local connectivity is a hyperparameter called the receptive field. Parameter sharing means weights connecting neurons to receptive fields are the same, namely, using the same filter to convolve the entire feature map. One reasonable assumption is that if one feature patch is useful for a region of an image, it will have the same impact on another region. So parameter sharing not only dramatically reduces the number of parameters but also leads to translation invariance which capture statistics in local patches, e.g., similar edges may appear at different locations. We denote $\mathbf{x}_i^l$ as the i^{th} input feature map of l layer, $\mathbf{k}_{ij}^l$ as the kernel connecting j^{th} feature map of the output layer to i^{th} feature map of the input layer and b_j^l as an additive bias. Hence we have that

$$\mathbf{x}_j^l = f\left(\sum_i \mathbf{x}_i^{l-1} * \mathbf{k}_{ij}^l + b_j^l\right) \tag{1}$$

where f is an activation function which is usually a rectified linear function.

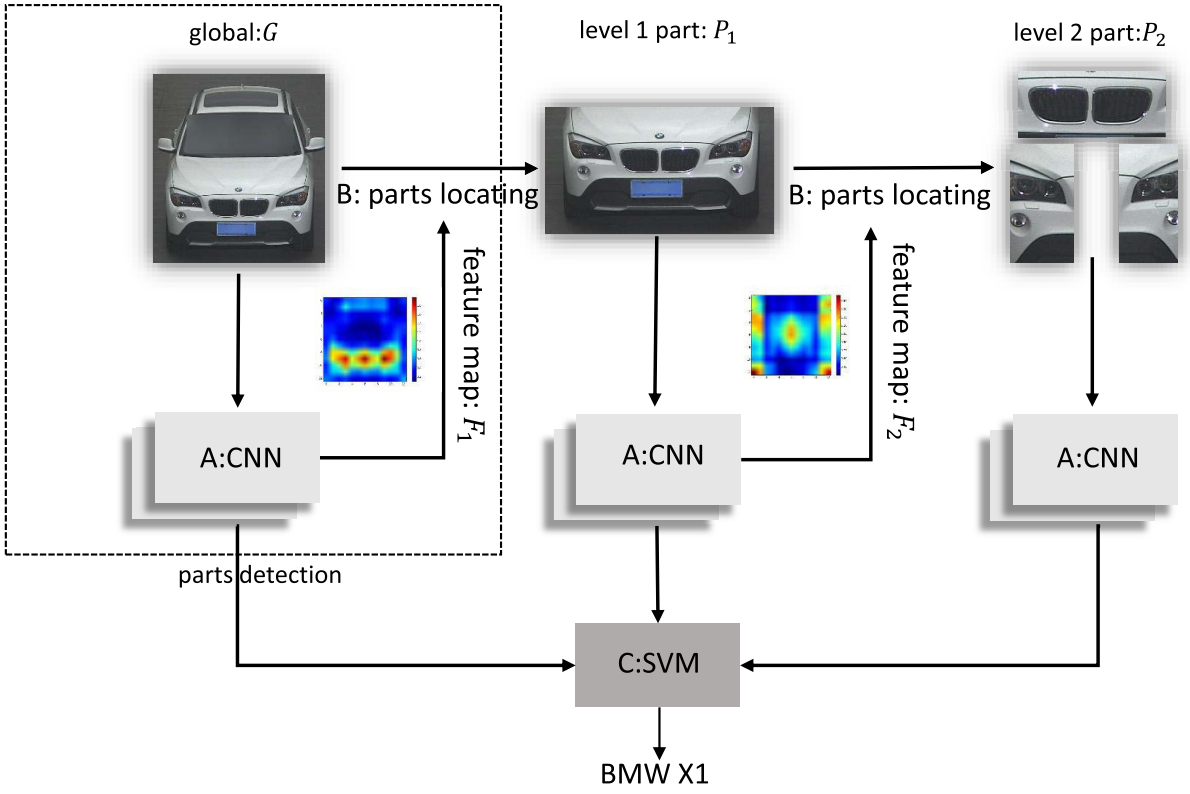

Fig. 1. The framework of our fine-grained vehicle model recognition. In general, there are three procedures, A, B and C. Procedure A is to train a CNN. Procedure B is to locate part regions utilizing feature maps generated by trained CNN. Procedure C is to train a SVM classifier. Our parts detection is surrounded by dash line where the global image set G is fed into a CNN and generate an average feature map F_1, then using our locating scheme to obtain level 1 part set P_1. The parts detection is showed more concretely in Fig. 2, and is also applied on set P_1 to obtain more subtle part set P_2. The detecting will keep running until there are no part regions to obtain. Finally, we combine the features of global and local regions to classify models using SVM.

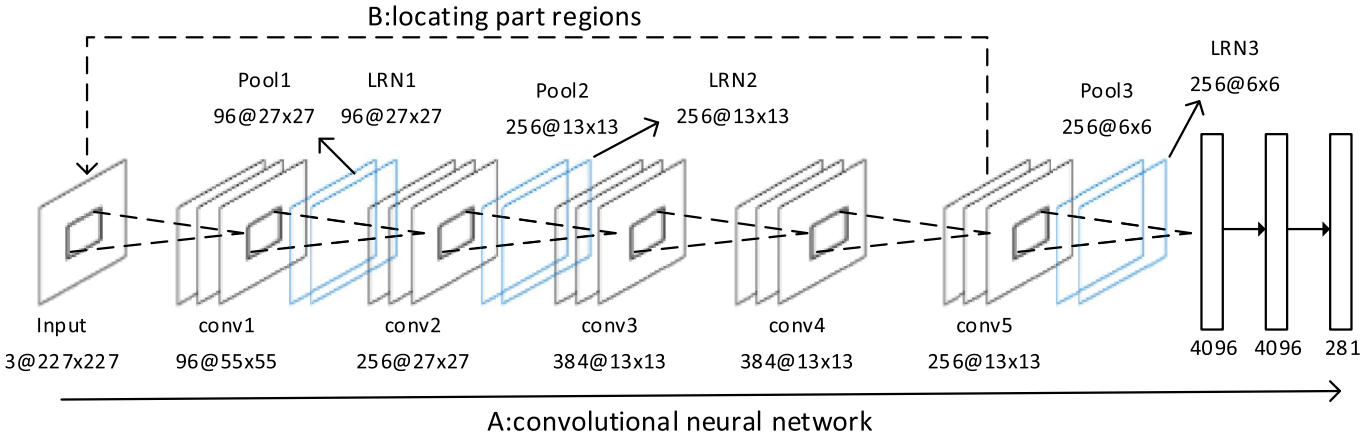

Fig. 2. The architecture of our parts detection. It primarily consists of training a CNN and locating part regions. The solid and dash line represents the A and B procedure described in Fig. 1 respectively. Both the procedures are established on a CNN. The network contains five convolution layers, three pooling layers, three local response normalization layers and three fully connected layers. The number in front of @ is the number of feature maps which is the output of corresponding layer. Since the input is an RGB image, it has three feature maps. The number behind @ is the size of feature maps. In the fully connected layers, the number represents the length of the feature vector, and the final layer is a softmax layer whose output number indicates the amount of fine-grained categories(e.g., 281 in our experiments). The feature maps of conv5 will be used in B procedure to locate regions of interest.

2) Pooling Layers: Pooling layers spatially combine convolution layers' outputs, which is related to classical spatial pyramid matching [22]. Its function is to dramatically reduce the spatial size of the feature maps as well as the amount of the parameters leading to efficiently computation in the network, and therefore also control overfitting. Besides reducing

computation in the network, max pooling, a most useful pooling method in CNN also provides a form of translation invariance. Formally denoted as:

$$\mathbf{x}_j^l = down\left(\mathbf{x}_j^{l-1}\right) \tag{2}$$

where $down(\cdot)$ is a kind of sub-sampling function.

3) Local Response Normalization Layers: Local response normalization layers give better generalization. Denoting by a_i the single value of i^{th} feature map, the response-normalized activity b_i is given by the expression

$$b_i = \frac{a_i}{\left(k + \alpha \sum_{j=max(0,1-n/2)}^{min(N-1,i+n/2)} a_j^2\right)^{\beta}} \tag{3}$$

The constants k, n, α, and β are hyperparameters, and we use $k = 2$, $n = 5$, $\alpha = 10^{-4}$, and $\beta = 0.75$ in our experiments.

4) Fully Connected Layers: Several fully connected layers at the end act as nested linear classifiers. The classifier most used in the final layer is softmax. The input of the softmax classifier is the feature vector with fixed dimensions, which is the output of the previous layers, and the output of softmax are the probabilities of the categories, in which the highest one is corresponding to the predicted category. For a binary classification using logistic regression, the hypothesis is

$$P(y = 1|\mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp\left(-\mathbf{w}^T\mathbf{x}\right)} \tag{4}$$

where y is label, $\mathbf{x} \in \mathbf{R}^{(D+1)\times 1}$ represents the D dimensional feature vector, $\mathbf{w} \in \mathbf{R}^{(D+1)\times 1}$ represents the weight vector. Considering a classification problem where the response variable y can take on any one of N values, we can generalize the binary classification, thus

$$P(y = c|\mathbf{x}; \mathbf{W}) = \frac{\exp\left(\mathbf{w}_i^T\mathbf{x}\right)}{\sum_{i=1}^{N} \exp\left(\mathbf{w}_i^T\mathbf{x}\right)} \tag{5}$$

where $\mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N] \in \mathbf{R}^{(D+1)\times N}$, each $\mathbf{w}$ represents the corresponding category weight parameters.

B. Locate Part Regions

One of our innovations is that we can automatically locate the important regions which contribute more for recognition, i.e., more discriminative among these regions. Because the last convolution layer extracts more discriminative and semantic features, we use feature maps of the last convolution layer to locate interesting parts, and this procedure is marked with dash line in Fig. 2. There are 256 feature maps with 13 × 13 size and we simply calculate the average value of these maps to achieve a new feature map, i.e., each pixel value in this new feature map is the mean of the corresponding pixel in the 256 feature maps. As demonstrated in Fig. 3, we use this process to all training images to obtain the average feature map of the entire training set and this procedure can be formalized as below:

$$F = \frac{\sum_{i=1}^{\#training} \frac{1}{256} \sum_{j=1}^{256} f_i^j}{\#training} \tag{6}$$

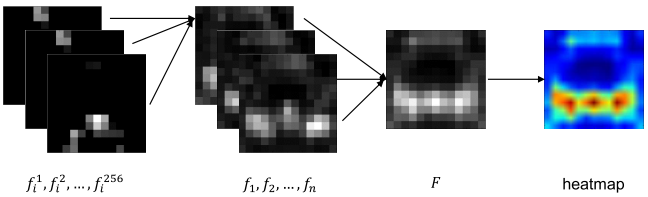

Fig. 3. The process of generating average feature map of the entire training data. f_i^j is the j^{th} feature map of the fifth convolution layer of the i^{th} image. f_i is the average fifth convolution layer's feature map. F is the average feature map of the entire training data. The heatmap is generated from F.

where f_i^j represents the j^{th} feature map of the fifth convolution layer of the i^{th} image. $\#training$ is the amount of the training set.

A heatmap of such an average map is generated to indicate which regions contribute more to the final prediction, i.e., the regions with brighter color are considered to be more discriminative for recognition among similar categories. The principle for selecting these regions is to choose the tight rectangular blocks which are obviously isolated and surround regions with brighter color. Of course these rectangular blocks are fairly smaller than the whole heatmap, otherwise we are not necessary to obtain the finer parts and can absolutely use the entire heatmap. If we get such a heatmap that has no other regions to obtain using our principle, we will not search for the finer parts. We use these extracted regions from the heatmap to obtain the corresponding areas of the input image. For the sake of this purpose, a mapping between the point in the heatmap and the input image should be established, which is obtained by the following formulations for the convolution and pooling layers:

$$p_i = s_i \cdot p_{i+1} + \left(\frac{k_i - 1}{2} - m_i\right) \tag{7}$$

and for other layers:

$$p_i = p_{i+1} \tag{8}$$

where p_i is the point in an input map of the i^{th} layer, p_{i+1} is the point in an input map of the $i + 1^{th}$ layer, and this map is also the output map of the i^{th} layer, s_i is the stride step, k_i is the kernel size, m_i is the padding length. Besides the convolution and pooling layer, the input map and output map have the same size, and the mapping scheme can be described by Eq. 8. The detail of the Eq. 7 is visualized in Fig. 4. In this example, m_i is 1, s_i is 2 which represents the convolution kernel slides two pixels between every calculation and the kernel size is 3×3. We regard the center of the area which is convolved by a convolution kernel as the mapping point of the convolutional result. Thus, the point p_{i+1} of the $i + 1^{th}$ layer has its corresponding point p_i of the i^{th} layer, the mapping point of the point p'_{i+1} is p'_i. And this mapping relation can be described by the Eq. 7.

To obtain the region in the input image, we can map the top-left corner and the bottom-right corner of the hot region in the heatmap. As illustrated in Fig. 4, our mapping scheme maps a point to the center of a field whose size is the kernel size, and our mapping is from the fifth convolution layer to the input image, if we only use every center point layer by layer, it will

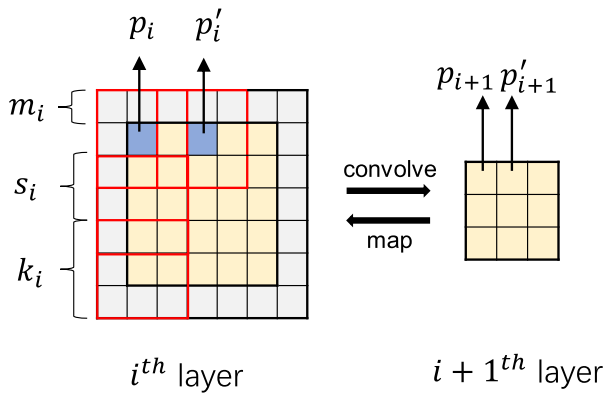

Fig. 4. The illustration of the mapping scheme.

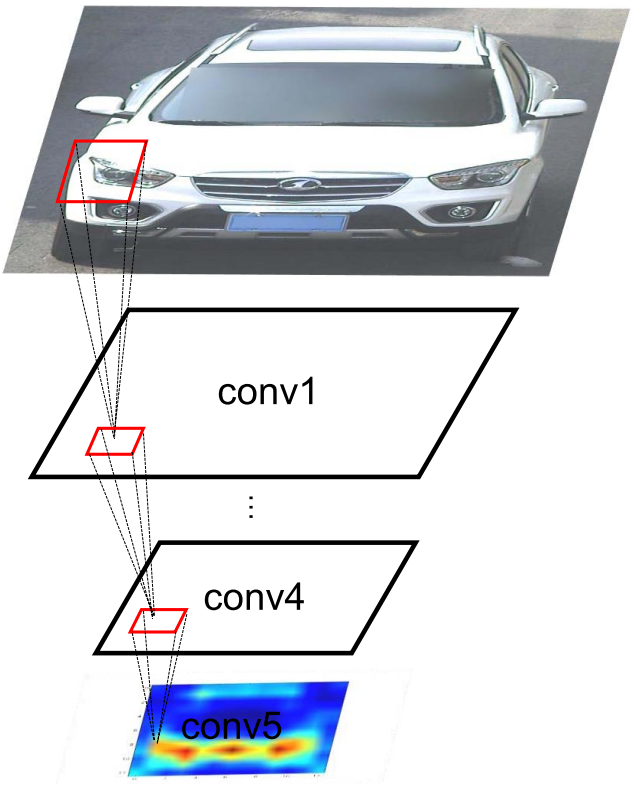

Fig. 5. The illustration of the overall mapping procedure.

result in a very tight region in the input image and lose most of other important information in the peripheral region. If we extend this part larger, of course we get a larger region which contains adequate information, but needs more iterations to obtain the final finest parts of the image. To solve this problem, we extend two points mapped into the input image K pixels where K is about a half of the summation of each kernel size and is set to 15 in our experiments, in other words, if the point (x_1, y_1) and the point (x_2, y_2) are the mapping points of the top-left corner and bottom-right corner, then the extended points are (x_1-15, y_1-15) and (x_2+15, y_2+15). The overall mapping procedure is illustrated in Fig. 5.

C. Coarse-to-Fine Detection

As illustrated in Fig. 1, the *parts detection* is composed of *CNN training* and *parts locating*. And the results of

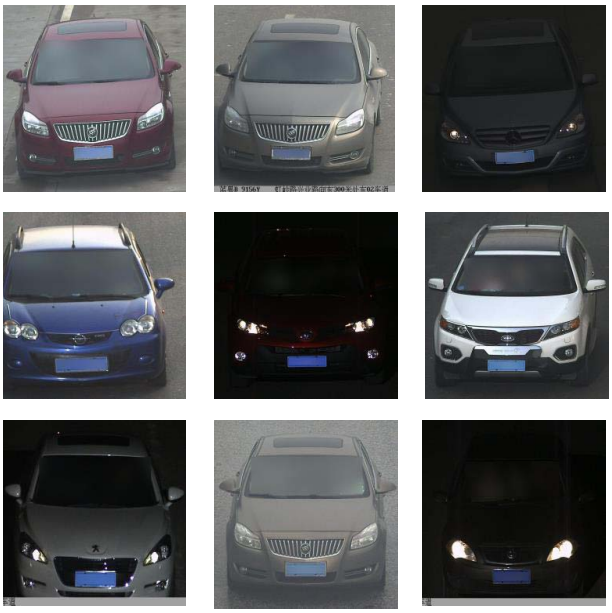

Fig. 6. Sample images of the surveillance-nature data. Images are affected by large variation from lightings and haze, and some images are little slant.

part detection will be regarded as the new input images, and be trained by the CNN again. Also we will locate the significant parts after training, and these parts are more subtle than the previous located parts. This procedure of detecting finer parts will keep running until we can't find the qualified regions using our selecting principle from a heatmap. We will use the final finest parts and the whole images to extract features for recognition.

D. One-Versus-All Linear SVM

We make use of an one-versus-all linear SVM to identify the fine-grained classes with the features extracted from the second fully connected layer. Given the training data $\{\mathbf{x}_i, y_i\}_{i=1}^n$, a SVM classifier is learned by minimizing the loss function

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \ \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i \\ &\text{subject to } y_i\left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \geq 1-\xi_i, \\ &\qquad \xi_i \geq 0. \end{aligned} \tag{9}$$

where $\mathbf{x}_i \in R^m$ and $y_i \in \{1, -1\}$. The hyperparameter $C > 0$ controls the penalty of the error terms. $\mathbf{x}_i$ are mapped into a higher dimensional space by function ϕ, in which SVM finds a linear separating hyperplane with the maximal margin.

In our approach, we use the one-versus-all linear SVM, so in the training stage, N SVM models are trained to classify N fine-grained vehicle categories. In each of the SVM models, the ground truth label is positive when it is the corresponding category, other category labels are denoted as negative. In the test stage, each SVM model takes on each test example respectively, calculate the output probability of each category. We regard the model which has the highest probability as the corresponding category.

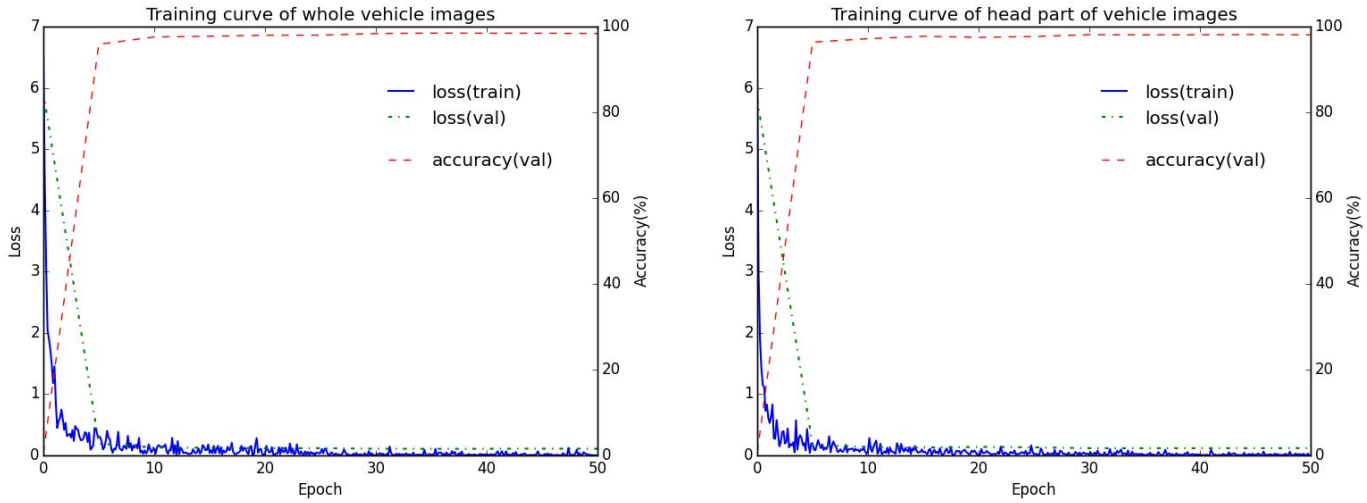

Fig. 7. Training curve of whole vehicle images and head part of vehicle images. The blue solid line represents the training loss, and the green dash-dotted line represents the validation loss. After 20 epochs, the green line lies above of the blue line which demonstrates our model is effective according to machine learning theory. The red dashed line represents the accuracy. The accuracy exceeds 90% at the first evaluation due to pre-trained model.

III. Experiment

The training stage of the CNN is known to be very time-consuming due to the parameters updating, even with the currently developed GPU technology. In the ITS domain, Huang *et al*. [23] used a PCA-based pretraining strategy to reduce the computational cost of training procedure in the vehicle logo recognition task. But each layer was pretrained with an unsupervised learning algorithm, which is not convenient and discriminative enough for fine-grained vehicle model recognition. Thus in our experiments, to speed up the training procedure, we use a pre-trained Caffe [20] model trained on a large auxiliary dataset (ILSVRC2012 classification) using image-level annotations only, then fine-tune the pre-trained model using our fine-grained vehicle model dataset to adapt to our specific task and domain, i.e., we initial our CNN weight parameters with the parameters of the pre-trained CNN model, then train the CNN to update the parameters using our own dataset. There have been a lot of works [24]–[26] using this pre-training strategy in the computer vision tasks. And there also have been literatures which try to research the theory behind this strategy, like [27]–[29]. As a result of using a pre-trained Caffe model, training a CNN only needs about 40 minutes in our experimental environment with a GeForce GTX TITAN X GPU and an Intel(R) Core(TM) i7-4790K CPU.

A. Dataset Description

We use the surveillance-nature data captured in the front view in the CompCars dataset [20] to train our CNN. Fig. 6 shows some examples of such surveillance images, which are affected by large variation from lighting and haze. The varying conditions, such as lighting, weather and so forth, make our fine-grained recognition more challenge but meaningful in realistic applications. On the other hand, some images, e.g., the final image in the middle row that is not exactly a frontal view but a little slanted, indicate our CNN is effective to some extent in a rotation situation. We use a total of 44481 images and split them into a training set and a test set without overlaps. The training set contains 281 vehicle models with a total of 31148 images and the test set also contains the same 281 vehicle models with 13333 images. In order to compute features by CNN, we first convert the images to a fixed 256×256 pixel size, as well as the detected contributive parts located by the feature map.

B. Training CNN

We extract deep convolutional features using an ImageNet pre-trained CNN model. In order to adapt to our specific domain and generate more discriminative features, we fine-tune the pre-trained CNN model. More specifically, we use a 281-way fc3 layer(the last fully connected layer), instead of a 1000-way fc3 classification layer, with randomly initialized weights drawn from a Gaussian with $\mu = 0$ and $\sigma = 0.01$. We initialize the global learning rate to a tenth of the original learning rate used by pre-trained model and drop it by a factor 10 after half training epochs, and initialize learning rate in the new fc3 layer by 10 times the global learning rate. In each Stochastic Gradient Descent(SGD) iteration, we use 100 images to construct a mini-batch. For each 5 iterations, we use the test set to calculate the accuracy. In the experiments, we eventually train 50 passes through the training data. We use the same training scheme on other parts of images. In fact, as shown in Fig. 7, training loss converges after 10 epochs, and because of the discriminative property of pre-trained CNN, the accuracy exceeds 90% at the first evaluation. This accuracy is calculated by the total number of correct predictions divided by the amount of test data:

$$accuracy_1 = \frac{\#correct\ predictions}{\#test\ data} \tag{10}$$

In addition, to demonstrate the effectiveness of our fine-tuned CNN, we also list the accuracy acquired by only using pre-trained model without fine-tuning in TABLE II.

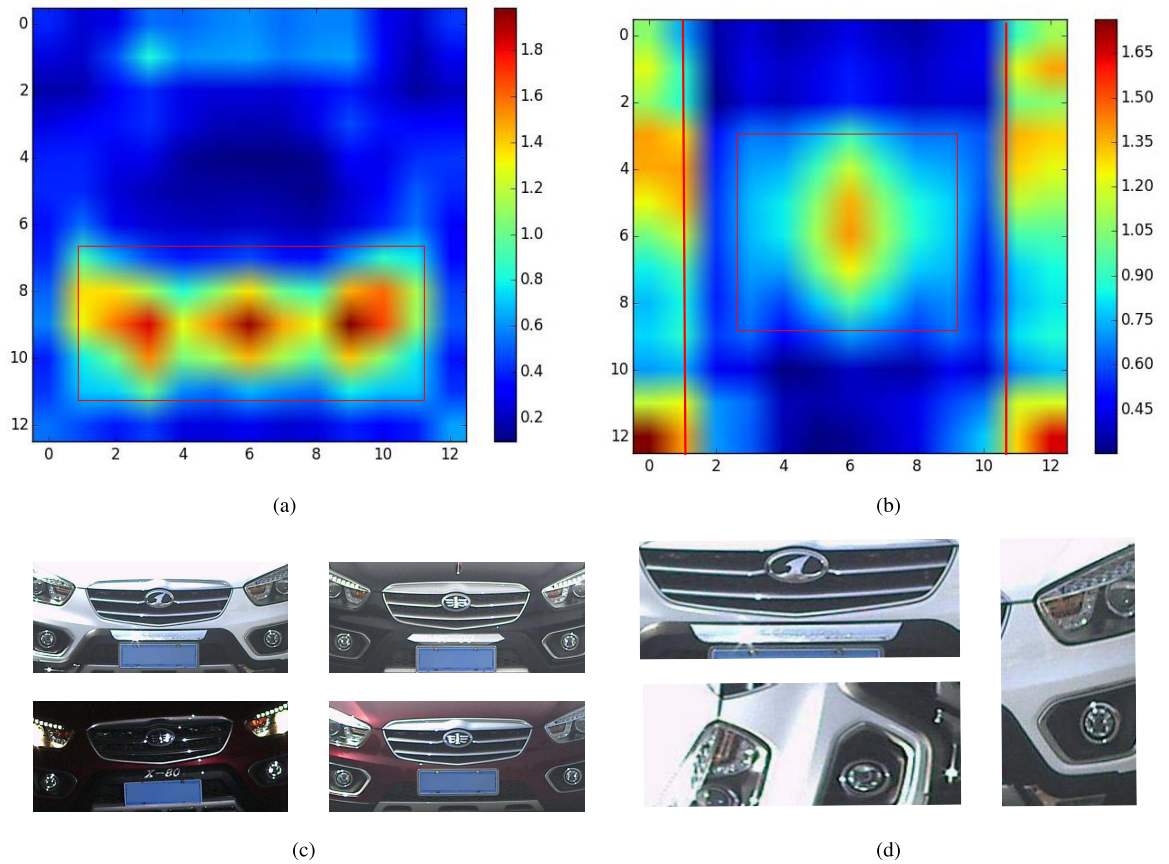

Fig. 8. Two example heatmaps of feature maps extracted from the whole and the head part of vehicle images, and its corresponding parts obtained via our locating scheme. Subfigure (a) is the heatmap of the whole vehicle images, the red rectangular region is selected using our selecting principle. These selected regions generate the corresponding parts depicted in the subfigure (c). The heatmap in the subfigure (b) is obtained using the parts in the subfigure (c) and its corresponding parts are illustrated in the subfigure (d). We rotate the left headlights 90 degree in order to make this figure tight.

C. Locating

Fig. 8 shows some example heatmaps of the fifth convolution layer on the different kinds of input images. Points $(7, 1)$ and $(11, 11)$ are selected from the heatmap of the whole vehicle image using our selecting principle in Fig. 8(a), and their corresponding parts are derived with the mapping scheme, shown in Fig. 8(c). Then we use these parts, all of which in the example are the head parts, to obtain another heatmap, as shown in Fig. 8(b). Based on the selecting principle, there are three hot regions in the heatmap, which are framed by red rectangles. We select points $(0, 0)$ and $(12, 1)$ for the left hot region, points $(3, 3)$ and $(9, 9)$ for the center hot region, points $(0, 11)$ and $(12, 12)$ for the right hot region. These coordinates will be applied on the test set to extract the corresponding parts. As shown in Fig. 8(d), center hot region is mapped into the center of head part of the vehicle, which is around vehicle logo. We also locate left and right headlights according to the left and right hot region. All these parts are detected automatically, and accordant with empirical method for recognizing fine-grained vehicle models. In our experiments, no other finer parts are utilized because there are no more other hot regions obtained from the heatmaps generated by parts in Fig. 8(d), i.e., our coarse-to-fine detection procedure finds the final finest parts in our situation.

D. Training SVM

After we get trained CNN models on the whole and finest parts of vehicle images, we use them to extract features of the second fully connected layer respectively. Depicted in Fig. 2, the output of the second fully connected layer is a 4096 dimensional feature vector, so we get such vectors from the whole and parts for each image, then concatenate them to a fixed dimensional feature vector. These fixed length feature vectors are utilized as the input to train the SVM model. The popular library for SVM, i.e., LIBLINEAR(https://www.csie.ntu.edu.tw/~cjlin/liblinear/) is used in our experiments. We set type of solver is 1 in LIBLINEAR. In order to get the best hyperparameter C in Eq. 9, we search a list of values of C using 5-fold cross-validation to prevent the overfitting problem. The cross-validation results of different C value are illustrated on TABLE I. Thus we set C to 0.6, the value with the highest accuracy, to train SVM using the entire training data.

E. Evaluation

For our experiments, we don't use $accuracy_1$ in Eq. 10 to evaluate our model's performance. The reason is that our surveillance-nature data in each category are not distributed uniformly. The TABLE V shows the amounts of

TABLE I
SELECT C USING 5-FOLD CROSS-VALIDATION

C	0.5	0.6	0.7	0.8	0.9	1
CV accuracy	98.88%	**98.90%**	98.88%	98.83%	98.866%	98.88%
C	1.1	1.2	1.3	1.4	1.5	1.6
CV accuracy	98.83%	98.82%	98.86%	98.85%	98.86%	98.88%

TABLE II
ACCURACY OF OUR EXPERIMENTS

Methods	Accuracy
whole(ft)	90.78%
head part(ft)	96.54%
whole + head part(ft + SVM)	97.98%
center(ft)	95.48%
left(ft)	93.69%
right(ft)	94.40%
whole + center + left + right(SVM)	95.56%
whole + center + left + right(ft + SVM)	**98.29%**

some categories in the training set and test set. The amount of the first category is the smallest and the last category contains the biggest. It is obvious that we should change another metric to evaluate our experiments. Because the $accuracy_1$ will be high enough as long as the total number of correct predictions is large, it will ignore the bad performance of some categories which have smaller number of correct predictions. Our dataset contains 281 models with large variation on the amount of each category, thus we must concentrate on the performance of each category, not on the performance of the entire dataset. For this purpose, we simply use an average accuracy to evaluate the performance:

$$accuracy_2 = \frac{\sum_i^N \frac{t_i}{n_i}}{N} \tag{11}$$

where N represents the number of categories, t_i represents the number of correct predictions on i^{th} category and the n_i is the amount of the i^{th} category. This metric can be regarded as the mean of the each category's accuracy calculated by the Eq. 10.

F. Results and Analysis

1) Results: We illustrate the experimental results of our approach and compare them with other fine-grained vehicle model recognition experiments. TABLE II shows the accuracy calculated by our formulation in which ft represents fine-tuned, i.e., the trained model utilized the pre-trained CNN model. We achieve 90.78% with fine-tuned CNN model for the whole vehicle images. It's much lower than the accuracy which is represented in Fig. 7. This indicates our evaluation method is reasonable to assess results on nonuniform dataset. The second row shows much higher accuracy only using head parts of vehicles as the training data. This result demonstrates our coarse-to-fine detection procedure locates more discriminative parts for recognition. The accuracy in the third row indicates that combining the global features and features of head parts achieves better performance. Then we detect more subtle regions and test these regions independently, the results are illustrated in the forth to sixth row respectively. And finally, our proposed method, which utilizes both the global and most subtle local features for recognition, achieves the best accuracy which reaches 98.29% as shown in the last row. This result is so remarkable that makes our method is able to be applied on the similar realistic tasks of ITS involving fined-grained vehicle model recognition. We also replace the fine-tuned CNN with pre-trained CNN to compare the performance between them, i.e., we use the pre-trained CNN to extract the features of the second fully connected layer. The result in the seventh row shows fine-tuned CNN can improve the performance in a specific task. Because the similar literatures of fine-grained vehicle model recognition used different evaluation method and fewer vehicle models, it is almost impossible to make an exactly fair comparison. Nevertheless, listing other works' results can give different views to analyze our work. Other relevant works are illustrated in TABLE III. These works' accuracies was calculated by Eq. 10, and we also list the accuracies of some literatures measured by Eq. 11 according to the data illustrated in these literatures. And as for other works, we can't find data in the paper to apply the Eq. 11, and marked with / symbol. Psyllos *et al.* [16] achieved 85% accuracy on the 11 models using a probabilistic neural network as a classifier, and got 88.59% accuracy assessed by Eq. 11. Petrovic and Cootes [30] achieved 93% accuracy on the 77 models using only frontal views of vehicles by locating, extracting and recognizing normalized structure samples taken from a reference image patch on the front of the vehicle. Clady *et al.* [14] used oriented-contour point based voting algorithm reports a maximum performance of 93.1% on the 50 vehicle classes by using fusion of different classifiers. Lee [15] employed a neural network trained with texture descriptors derived from the front view images to classify 24 types of Korean vehicles and obtains 94% recognition rate. He *et al.* [17] obtained 92.47% accuracy on the 30 models using an ensemble of neural network classifiers. Llorca *et al.* [18] achieved 94% accuracy on the 52 models but got 92.21% accuracy calculated by our evaluation method. Zhang [13] achieved 98.65% accuracy using his two-stage classification scheme, but only on the 21 models, and changed a little using Eq. 11, achieved 98.69%. Hsieh *et al.* [31] achieved 99.07% accuracy but decreased to 97.96% assessed by Eq. 11. Compared with these works, our method achieves a state-of-the-art accuracy on the 281 models, which exhibits promising potential for real-world application. As illustrated in TABLE III, [13] and [31] obtained the results as competitive as ours, for a fair comparison, we also implement their methods

TABLE III
REPORTED RESULTS OF SOME STATE-OF-THE-ART METHODS

Methods	#classes	Accuracy1	Accuracy2
Psyllos et al.[16]	11	85%	88.59%
Petrovis and Cootes[30]	77	93%	/
Clady et al.[14]	50	93.1%	/
Lee[15]	24	94%	/
He et al.[17]	30	92.47%	/
Llorca et al.[18]	52	94%	92.21%
Zhang[13]	21	98.65%	98.69%
Hsieh et al.[31]	29	99.07%	97.96%
Ours	281	98.63%	98.29%

TABLE IV

COMPARISONS OF OUR METHOD WITH OTHER STATE-OF-THE-ART METHODS ON THE COMPCARS DATASET

Methods	Zhang[13]	Hsieh et al.[31]	Ours
Accuracy	83.78%	51.70%	98.29%

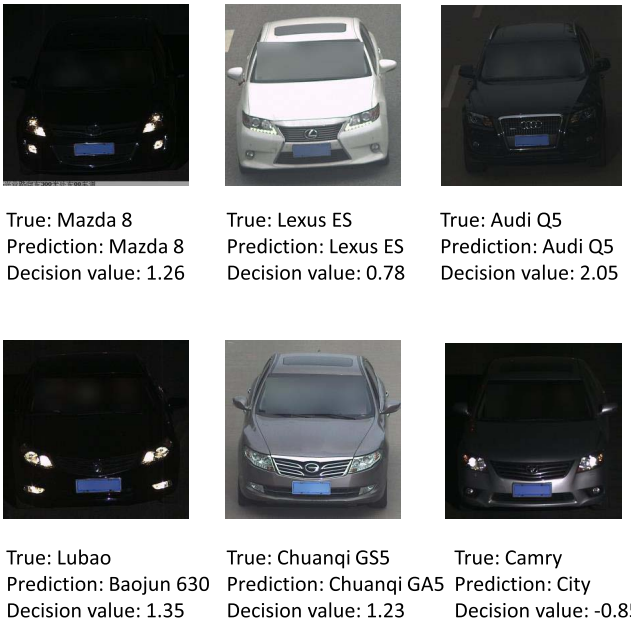

Fig. 9. Sample images of prediction results. Its ground truth, prediction and decision value calculated by SVM classifier are listed below.

on the CompCars [20]. The results are illustrated in TABLE IV, we achieved 83.78% accuracy using the method of [13] and 51.70% accuracy using [31]'s method. Except for the lower accuracy, both the methods have some limitations in the training stage when the dataset is large-scale. The method [13] needs several days to extract features of the entire dataset, and both of them cost much time on selecting parameters of classifiers in the training stage. These results demonstrate our proposed method is more efficient, accurate and convenient on the large-scale dataset which contains hundreds of models like CompCars [20].

2) Results Visualization: We illustrate some typical prediction examples in Fig. 9, in which the examples in the first row are predicted correctly and the examples in the second row are wrong predictions. To further investigate the experimental results, we collect our all wrong predictions and make a statistic. 87% wrong predictions are predicted to be another category with the same make like the middle image in the second row of Fig. 9, and about 7% wrong predicted images are in extremely low lighting conditions. However, even in such a extraordinary lighting condition, our approach can also make a prediction in the most cases like the first image in the first row of Fig. 9.

As shown in TABLE III, our method utilizing both the whole and finest parts of images gets the highest accuracy than the one that just uses the whole vehicle images or parts of images. We are also interested in the question that does

Fig. 10. Images in another dataset different from ours to test generalization of our trained model. These images are taken by different traffic cameras.

our method improve the accuracies of categories which have low accuracy? In fact, improving the accuracies of categories that have weak prediction is very useful in realistic application which makes the system more robust. Thus, we compare the accuracies of the lowest 10 categories obtained by using the combined features of whole images and the head parts with the accuracies for the same categories but for features obtained by combining the whole vehicle image and three parts. The results are shown in TABLE VI, Citroen C4, Venucia D50, Crider, encore, Venucia R50 improve their accuracies, especially all test images of RIO Class are predicted correctly. These improvements demonstrate the effectiveness of our method that can rise the performance of the entire system. We find the reason for the much lower accuracies of Ruiping and Volkswagen is that the amount of them is small and about half of these images are in dark condition, which results in less information for recognition.

3) Generalization: We also apply our trained model on another dataset, as shown in Fig. 10. These images are captured in different viewpoints and distances from cameras compared with CompCars. We test 160 images containing 9 categories. These 9 categories are Audi A7, Benz R Series, BMW X1, BMW X6, BYD-F6, BYD-S6, Mazda 8, Mazda CX7 and Volvo XC60. In order to test generalization of our model, we not only select different vehicle models with different vehicle makes but also with the same vehicle makes. The result shows that our approach will be somehow affected by the difference between the viewpoints of the training images and the test images, because such a difference will lead to wrong localization. However, the test accuracy of the new images still reaches 90%, that indicates our trained model has good generalization to other dataset and generates the competitive results. The amount and accuracy of each vehicle model are illustrated in TABLE VII.

TABLE V
AMOUNTS OF SOME CATEGORIES IN TRAINING SET AND TEST SET

vehicle model	RIO	Antara	Weizi	Cross Lavida	Changan CX20	Lexus RX	Audi Q5	Freelander	Teana	Heyue	Jetta
amount in training set	14	19	24	34	99	164	216	261	410	496	565
amount in test set	6	8	10	15	43	70	93	112	176	213	242

TABLE VI
IMPROVEMENT ON LOW ACCURACY CATEGORIES USING OUR PROPOSED METHOD

vehicle model	Ruiping	Volkswagen CC	Citroen C4	RIO	Venucia D50	Crider	encore	Cross Lavida	Octavia	Venucia R50
whole+head part(ft+SVM)	0.571	0.571	0.625	0.833	0.851	0.857	0.857	0.867	0.882	0.884
whole+center+left+right(ft+SVM)	0.571	0.571	**0.875**	**1**	**0.915**	**0.929**	**0.929**	0.867	0.882	**0.897**

TABLE VII
AMOUNT AND ACCURACY OF EACH VEHICLE MODEL IN OUR SMALL DATASET

vehicel model	Audi A7	Benz R Series	BMW X1	BMW X6	BYD F6	BYD S6	Mazda 8	Mazda CX7	Volvo XC60
amount	10	21	14	18	27	22	15	22	11
accuracy	1	0.810	0.857	0.722	0.963	0.818	0.933	1	1

IV. CONCLUSION

We propose a coarse-to-fine framework for fine-grained vehicle model recognition. Our method combines the global features and significant local features, which improves the performance on recognition for subordinate-level categories that have extremely similar appearance, and achieves the state-of-the-art results. The results show that significant regions which are the most discriminative regions among these fine-grained categories can improve the accuracy of recognition. And an automatically detecting procedure can exactly obtain such regions more effectively than the ones obtained by human's extraction. Such automatical procedure can also improve the generalization of the entire system. In the future work, we will focus on the training data with different viewpoints. It is a quite challenge problem, such work can be utilized to solve more complicated situations.

REFERENCES

[1] S. M. Khan, H. Cheng, D. Matthies, and H. Sawhney, "3-D model based vehicle classification in aerial imagery," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 1681–1687.

[2] W. Wu, Z. QiSen, and W. Mingjun, "A method of vehicle classification using models and neural networks," in *Proc. IEEE VTS 53rd Veh. Technol. Conf. (VTC Spring)*, vol. 4. May 2001, pp. 3022–3026.

[3] A. H. S. Lai, G. S. K. Fung, and N. H. C. Yung, "Vehicle type classification from visual-based dimension estimation," in *Proc. IEEE Intell. Transp. Syst.*, Aug. 2001, pp. 201–206.

[4] X. Ma and W. E. L. Grimson, "Edge-based rich representation for vehicle classification," in *Proc. 10th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2. Oct. 2005, pp. 1185–1192.

[5] Y. Peng, Y. Yan, W. Zhu, and J. Zhao, "Vehicle classification using sparse coding and spatial pyramid matching," in *Proc. IEEE 17th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2014, pp. 259–263.

[6] M. C. Narhe and M. Nagmode, "Vehicle classification using SIFT," *Int. J. Eng. Res. Technol.*, vol. 3, no. 6, Jun. 2014, pp. 1735–1738.

[7] A. P. Psyllos, C.-N. E. Anagnostopoulos, and E. Kayafas, "Vehicle logo recognition using a SIFT-based enhanced matching scheme," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 322–328, Jun. 2010.

[8] D. F. Llorca, R. Arroyo, and M. A. Sotelo, "Vehicle logo recognition in traffic images using HOG features and SVM," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 2229–2234.

[9] H. Yang *et al.*, "An efficient method for vehicle model identification via logo recognition," in *Proc. 5th Int. Conf. Comput. Inf. Sci. (ICCIS)*, Jun. 2013, pp. 1080–1083.

[10] J. Prokaj and G. Medioni, "3-D model based vehicle recognition," in *Proc. Workshop Appl. Comput. Vis. (WACV)*, Dec. 2009, pp. 1–7.

[11] K. Ramnath, S. N. Sinha, R. Szeliski, and E. Hsiao, "Car make and model recognition using 3D curve alignment," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2014, pp. 285–292.

[12] J. Zhan, H. Zhang, and X. Luo, "Fine-grained vehicle recognition via detection-classification-tracking in surveillance video," in *Proc. 5th Int. Conf. Digit. Home (ICDH)*, Nov. 2014, pp. 14–19.

[13] B. Zhang, "Reliable classification of vehicle types based on cascade classifier ensembles," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 322–332, Mar. 2013.

[14] X. Clady, P. Negri, M. Milgram, and R. Poulenard, "Multi-class vehicle type recognition system," in *Artificial Neural Networks in Pattern Recognition*. L. Prevost, S. Marinai, and F. Schwenker, Eds. Berlin, Germany: Springer, 2008, pp. 228–239.

[15] H. J. Lee, "Neural network approach to identify model of vehicles," in *Advances in Neural Networks*. Berlin, Germany: Springer, 2006, pp. 66–72.

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

[17] H. He, Z. Shao, and J. Tan, "Recognition of car makes and models from a single traffic-camera image," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3182–3192, Dec. 2015.

[18] D. Llorca, D. Colás, I. Daza, I. Parra, and M. A. Sotelo, "Vehicle model recognition using geometry and appearance of car emblems from rear view images," in *Proc. IEEE 17th Int. Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2014, pp. 3094–3099.

[19] A. J. Siddiqui, A. Mammeri, and A. Boukerche, "Towards efficient vehicle classification in intelligent transportation systems," in *Proc. 5th ACM Symp. Develop. Anal. Intell. Veh. Netw. Appl.*, 2015, pp. 19–25.

[20] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3973–3981.

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

[22] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2. Jun. 2006, pp. 2169–2178.

[23] Y. Huang, R. Wu, Y. Sun, W. Wang, and X. Ding, "Vehicle logo recognition system based on convolutional neural networks with a pretraining strategy," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1951–1960, Aug. 2015.
[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.
[25] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.
[26] F. Liu, G. Lin, and C. Shen, "CRF learning with CNN features for image segmentation," *Pattern Recognit.*, vol. 48, no. 10, pp. 2983–2992, Oct. 2015.
[27] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.
[28] J. Donahue *et al.* (Oct. 2013). "DeCAF: A deep convolutional activation feature for generic visual recognition." [Online]. Available: https://arxiv.org/abs/1310.1531
[29] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2014, pp. 512–519.
[30] V. S. Petrovic and T. F. Cootes, "Analysis of features for rigid structure vehicle type recognition," in *Proc. BMVC*, 2004, pp. 1–10.
[31] J.-W. Hsieh, L.-C. Chen, and D.-Y. Chen, "Symmetrical SURF and its applications to vehicle detection and vehicle make and model recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 1, pp. 6–20, Feb. 2014.

Jie Fang (S'15) received the B.E. degree from the School of Electronic Science and Engineering, Nanjing University, China, in 2015, where he is currently working toward the M.E. degree. His research interests include intelligent transportation systems, computer vision, and machine learning and its application.

Yu Zhou (M'10) received the M.E. degree in circuits and systems and the Ph.D. degree in signal and information processing from Nanjing University, China, in 2005 and 2008, respectively. In 2012 and 2013, he was a Visiting Scholar with University of Kentucky, USA. He is currently an Associate Professor with the School of Electronic Science and Engineering, Nanjing University. His research interests include intelligent transportation systems, computer vision, and machine learning and its application.

Yao Yu (M'10) received the B.E. and Ph.D. degrees from Nanjing University, China, in 2005 and 2010, respectively. He was a Research Assistant with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong, from 2007 to 2008. He is currently an Associate Professor with the School of Electronic Science and Engineering, Nanjing University. His research interests include 3-D geometric modeling and computer vision.

Sidan Du (M'02) received the B.S. and M.S. degrees in electronic engineering from Xidian University, Xian, China, in 1984 and 1987, respectively, and the Ph.D. degree in physics from Nanjing University, Nanjing, China, in 1997. She is currently a Professor with the School of Electronic Science and Engineering, Nanjing University. Her research interests include in digital imaging processing and computer vision.