A Cascaded Part-Based System for Fine-Grained Vehicle Classification

Mohsen Biglari, Ali Soleimani, and Hamid Hassanpour

***Abstract*— Vehicle make and model recognition (VMMR) has become an important part of intelligent transportation systems. VMMR can be useful when license plate recognition is not feasible or fake number plates are used. VMMR is a hard, fine-grained classification problem, due to the large number of classes, substantial inner-class, and small inter-class distance. A novel cascaded part-based system has been proposed in this paper for VMMR. This system uses latent support vector machine formulation for automatically finding the discriminative parts of each vehicle category. At the same time, it learns a part-based model for each category. Our approach employs a new training procedure, a novel greedy parts localization, and a practical multi-class data mining algorithm. In order to speed up the system processing time, a novel cascading scheme has been proposed. This cascading scheme applies classifiers to the input image in a sequential manner, based on the two proposed criteria: confidence and frequency. The cascaded system can run up to 80% faster with analogous accuracy in comparison with the non-cascaded system. The extensive experiments on our data set and the CompCars data set indicate the outstanding performance of our approach. The proposed approach achieves an average accuracy of 97.01% on our challenging data set and an average accuracy of 95.55% on CompCars data set.**

***Index Terms*— Fine-grained classification, vehicle make and model recognition, cascading scheme, deformable part models, latent support vector machine (SVM).**

I. Introduction

VEHICLE analysis has attracted much attention recently. It is widely used in various vehicle-centered applications such as intelligent traffic and transportation systems, large-scale vehicle searches, intelligent parking, automatic toll collection and number-plate forgery detection. Vehicle Make and Model Recognition (VMMR) can complement Automatic Number Plate Recognition (ANPR) systems. For example, number-plates forging cannot be detected by using only APNR systems.

Vehicle analysis tasks can be divided into three main categories: vehicle detection, vehicle type recognition (i.e. bus, van, car, and motorcycle) and fine-grained vehicle classification. Notable progress has been made in the first two categories [1], [2]. In fine-grained vehicle classification, the main category of objects is known and the goal is the identification of the fine-grained category of the given vehicles. Compared to objects, vehicles have unique structural and visual properties, which poses a range of challenges to the field. Vehicles have a large intra-class variation and a small inter-class distance.

To address the above-mentioned challenges, a novel approach has been proposed for VMMR in this paper. Our approach is an upgraded version of the system presented in [3], as well as a new cascading scheme. In addition, we performed a larger set of experiments. We customized the model training procedure and parts localization algorithm. Moreover, a multi-class data mining algorithm for hard negative samples is used. The proposed training procedure employs a combination of standard support vector machine (SVM) and Latent SVM [4] algorithms to learn a model per class of vehicles, and finds the discriminative parts of each class simultaneously. Each model contains a root filter, part filters and the spatial relationship of parts. Root filter captures global appearances, and part filters capture local appearance properties of a specific vehicle class. The spatial relationship between parts helps a model to distinguish between similar classes with a different configuration of parts. Such a model can be learned using Latent SVM or MI-SVM algorithms [5]. These individual models are used together with a max-voting scheme. However, the models aren't calibrated and their scores can't be compared to each other as they are. Thus, we used Platt-Scaling [6] and fitted a logistic regression model to all of the classifiers scores. Furthermore, in order to decrease the system processing time, a novel cascading scheme has been proposed. This cascading scheme can control the trade-off between accuracy and speed of the system, by applying the classifiers in a sequential manner to the input. We proposed a practical ordering of classifiers based on two criteria, namely confidence, and frequency. For further boosting the system speed, each model in the cascaded system is replaced with its cascaded version. With this modification, the system processing time is reduced by 93%. For evaluation purposes, we used our dataset, namely BVMMR Dataset v2, which contains more than 5000 vehicles of 28 different makes and models [3] and comprehensive CompCars [7] dataset. The experimental results on these two datasets show the excellent performance of the proposed approach.

The rest of the paper is organized as follows: Section 2 reviews the related works. Section 3 describes the proposed approach. Section 4 gives the experimental results. Conclusions are finally made in Section 5.

II. Related Works

Almost all of the research papers reviewed in this section have used a private dataset to evaluate their approach, hence

Manuscript received December 2, 2016; revised February 15, 2017 and May 24, 2017; accepted July 2, 2017. The Associate Editor for this paper was Q. Wang. *(Corresponding author: Mohsen Biglari.)*

The authors are with the Department of Computer Engineering and IT, Shahrood University of Technology, Semnan 316, Iran (e-mail: mbt925@gmail.com; solimani_ali@shahroodut.ac.ir; h.hassanpour@shahroodut.ac.ir).

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.2017.2749961

they cannot be compared based on their reported results. Although many of them claimed that their accuracy is high, their datasets have neither a large number of images nor enough number of classes.

We divided the related works into two main categories: holistic approaches and part-based approaches. Holistic approaches use the whole image or a region of interest (ROI) for feature extraction. They try to find a global description of the image. While part-based approaches try to localize parts and utilize them for feature extraction and classification.

A. Holistic Approaches

Holistic approaches don't use any part-specific features and solely act based on a region of interest. So they are sensitive to noise, localization errors and small viewpoint variations. The main drawback of holistic approaches is the use of imperfect assumption for localizing ROI which can be seen in most of the related works in this category. For example, Conos [8] initially locates the number-plate, then uses some relative measures to determine ROI; dependency to number-plate calls one of the major application of VMMR into question. After ROI detection, a simplified version of SIFT and LDA have been used for feature extraction and k-NN as the classifier. Some other researchers used similar approaches for ROI localization [9]–[11].

Several single-class and multi-class classification approaches have been experimented in [12]. They used frontal lights in addition to ROI for feature extraction. Relative headlight positions are determined using morphological operators, then the distance from edges of ROI and headlights to the center of ROI are calculated and used as the feature vector. Boonsim and Prakoonwit [13] utilized a combination of salient geometric and shape features of taillight and license plate to cope with limited lighting conditions at night. They employed SVM, decision tree and k-NN using a majority-voting scheme.

A wide variety of feature extraction methods like edge orientation, Harris corner detection, Sobel edge response and Square Mapped Gradients have been tested in [14] and Square Mapped Gradients has been reported as the best one. Pearce and Pears [10] don't have the same opinion as [14]. They pointed out that if Harris corner detection was applied locally and in a multi-region manner, it would have outperformed square mapped gradients. However, these two works are evaluated on different datasets. Aggregate Channel Features (ACF) is applied to the whole image in [15]. They used multiple SVM classifiers with a weighting mechanism to classify traffic signs.

A novel symmetrical SURF was recently proposed in [16] to enrich the power of SURF descriptor to detect all possible symmetrical matching pairs through a mirroring transformation. ROI detection is done using the symmetric property of vehicles and without using any motion features. Afterward, the ROI is divided into 4×6 regions and the extracted features via Histogram of Oriented Gradients (HOG) and SURF from each region are used to train multiple weak classifiers. These weak classifiers are then integrated to build a strong ensemble classifier using a voting scheme.

B. Part-Based Approaches

The part-based approaches are in search of discriminative or meaningful parts in the image. For example, one can separate a large number of classes of vehicles by locating the lights and using them in classification. The significant parts of these approaches are indeed detection and localization of the parts. One of the shortcomings of part-based approaches is the use of static and imperfect assumption for localizing parts. For example, Psyllos *et al.* [17] considered logo as a discriminative part. They calculated phase congruency of the extracted ROI and filtered out the feature map values smaller than a threshold. They assumed that the peak in the central region of phase congruency map gradient is where the logo usually appears. This assumption is based on having grills with horizontal lines which do not apply to all makes and models. Yang *et al.* [18] used a similar approach. They concatenated features extracted from a region around the logo together with the logo itself.

Santos and Correia [19] used backlight edges, angles and their spatial position relative to the number-plate as the features. Emblems are considered as a discriminative feature in [20]. Sarfraz and Khan [21] used a probabilistic patch-based framework which automatically learns a discriminative patch set for each vehicle subcategory in the training phase. A class label is assigned to a query patch by finding its similarity with the patch set and utilizing prior models in a pure Bayesian framework. Finally, the class label of the query patch is determined by the maximum number of patches assigned to a given class. In a conceptually similar approach, individuals in crowd videos are classified into different groups by the motion of feature points and manifold learning [22]. Likewise, Wang *et al.* [23] segmented multiple pixels into a superpixel SLIC method and Context-aware Label Transfer. They utilized weakly supervised SVM classifier to classify each superpixel.

Convolutional neural networks (CNNs) are a powerful class of models for visual recognition. The use of CNNs has grown substantially due to the outstanding performance on a wide variety of vision tasks like object classification [24]–[26] and object detection [27], [28]. CNNs and Deformable part models (DPMs) are typically viewed as two distinctive approaches, but the work of [29] demonstrated that a DPM can be formulated as a CNN. Hence, we consider CNNs as a part-based approach like DPMs. There are a couple of recent VMMR approaches based on CNNs. For instance, [30] has compared the accuracy of classification of four vehicle types (Bus, Truck, Van and Small cars) using CNN and SVM. CNN slightly outperformed SVM on their private dataset. A Hierarchical joint CNN-based model is reported in [31]. They used a region proposal method and the hierarchical relations within the fine-grained categories to improve classification accuracy. Gao and Lee [26] passed the binary image of detected ROI to CNN and entrusted CNN to do the feature extraction itself. Recently, they improved their work and utilized HOG for feature extraction prior to CNN [32]. This modification results in a higher accuracy and half the speed of the baseline system. The 3D vehicle bounding box and its orientation are used for decreasing the classification error in [24]. They were able to reduce the classification error

by 26% and improve the accuracy by 6% via adding this extra information to their extracted feature vector.

Conceptually, our approach is similar to the works of [33]– [35]. But our idea for finding discriminative parts set is totally different from theirs. The main focus of [35] is on viewpoint. Their goal is to find some visually similar parts in a specific viewpoint. Sarfraz *et al.* [34] used a fixed set of patches for all vehicle classes. The main difference between our approach and these two methods is that neither of them used separate parts set for each class of vehicles. Recently, a coarse-to-fine CNN architecture has been proposed in [33]. They employed a similar part-based approach and found discriminative parts set for classification of different makes and models. However, in their work, the parts location are shared across all classes.

III. The Proposed Approach

There are some prior works which search for semantically meaningful parts* to humans in vehicle images. But necessarily, not every meaningful part has a visually discriminative power. For example, a part on the left of the hood of a car can be visually discriminative but doesn't have a name or a label. In this paper, we propose an algorithm for automatically discovering the discriminative parts set of each vehicle class using Latent SVM [4]. To the best of our knowledge, it has not been well investigated in the context of VMMR.

The proposed approach learns a model M_c per subcategory c from an automatically found discriminative parts set [3]. Each model $M_c = \{F_r, F_{parts}, S_{parts}\}$ includes a root filter, part filters and spatial relationships of parts. Such a model can capture global and local appearance properties of a specific subcategory. The spatial relationship of parts helps a model to distinguish between similar classes with different configurations of parts. Root filter is a rectangular template specifying weights for a window, placed on different locations in the input image. Part filters have a smaller size, placed within a root window. The spatial relationship defines a set of allowed placements for a part relative to a root window.

We divide a model M_c into filters (β) and spatial configuration (z), and learn these two set of parameters from labeled training data $D = (\langle x_1, y_1\rangle, \ldots, \langle x_n, y_n\rangle)$ where $\mathrm{x_i}$ specifies the image and $y_i \in \{1, -1\}$ represents the label, with minimizing the following objective function [3]:

$$L_D(\beta) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{n}\max(0, 1 - y_i f_\beta(x_i)) \quad (1)$$

$$f_\beta(x) = \beta \cdot \phi(x, z) \quad (2)$$

$f_\beta(x)$ is a classifier which scores a sample x with respect to a specific vector of model filters β, and a spatial configuration z. $\phi(x, z)$ returns the feature vector of sample x when placing the filters according to the configuration z. The set $Z(x)$ defines the possible latent values of z for a sample x. The first term in the summation of (1) encourages increasing the margin size and the second set of terms (standard hinge loss) ensures that the sample x_i lies on the correct side of the margin. The constant C controls the trade-off between these two terms. We can obtain a strong model with an iterative training algorithm on (1) that alternates between fixing latent values z for positive examples and optimizing the Latent SVM objective function.

* A part with a label, i.e. lights and grilles

Procedure Learn

Function $\mathrm{M_c}$ = **Learn**(Class c)
$\mathrm{F_r}$: $[0]_{\mathrm{R_1 \times R_2}}$
pos: positive samples (vehicle class c)
neg: subset(negative non-vehicle samples, random)
vehNeg: subset(negative vehicle samples, K-uniform)
1. **Train**($\mathrm{F_r}$, pos, neg, 1, 1, SVM)
2. **Train**($\mathrm{F_r}$, pos, subset(vehNeg, L-uniform), 4, 3, SVM)
3. Initialize parts filters ($\mathrm{F_{parts}}$) and positions ($\mathrm{S_{parts}}$) using $\mathrm{F_r}$
$\mathrm{M_c} = \{\mathrm{F_r}, \mathrm{F_{parts}}, \mathrm{S_{parts}}\}$
4. **Train**($\mathrm{M_c}$, pos, subset(vehNeg, L-uniform), 8, 10, LSVM)
5. **Train**($\mathrm{M_c}$, pos, vehNeg, 1, 5, LSVM)
end

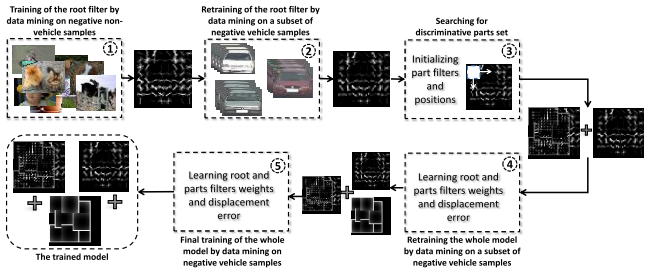

Fig. 1. The steps used by the proposed system to train a model for a specific class of vehicles.

A. Training a Model

We extended the approach proposed in [4] to cope with a multi-class problem. We customized the training procedure, the initialization for the locations of the parts and the data mining procedure for hard negative samples. The following is the proposed training procedure that learns the model M_c for vehicle class c. The flow diagram of the training procedure is represented in Figure 1.

$R_1 \times R_2$ is the dimension of the root filter. Our proposed multi-class version of data mining hard negative examples includes three sets; neg and $vehNeg$ which are two separate negative sets and pos is the positive set for class c. neg set contains non-vehicle images and $vehNeg$ contains vehicle images from classes other than the class c. A key factor to gain a good model is the way $vehNeg$ set is selected. We experimented two methods for the selection of $vehNeg$ set: random selection and K-Uniform selection. Uniform selection led to the best performance. In this method, an equal number of samples (K) from each of the $M-1$ class (other than class c) are included in the $vehNeg$ set. The size of neg and $vehNeg$ (K and L) sets are selected based on the dataset size, the total number of classes and the computer's RAM size. In the experiments, we set the size of neg and $vehNeg$ each equal to 5000. In BVMMR dataset with 5000 images, this $vehNeg$ size means all non-vehicle negative samples, but in CompCars dataset, this means nearly 30 samples per

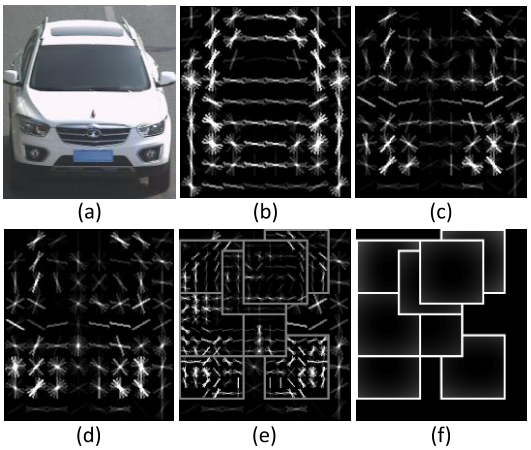

Fig. 2. The root filter for "Besturn x80 366" class of CompCars dataset (a) after step 1 (b) and step 2 (c). The filters specify weights for HOG [36]. The filters visualization show the positive weights at different orientations. The fully trained model is shown in the second row. Root filter (d), part filters (e) and a spatial model for the location of each part relative to the root (f). The visualization of the spatial models displays the cost of placing the center of a part at different locations relative to the root.

class (SPC). In other words, we set K = 30 and L = 10 for CompCars dataset.

The Train procedure which is presented below takes a model/filter, a positive set, a negative set, the number of training iterations and the kind of training procedure as inputs and returns the trained model. Steps 1 and 2 train just the root filter using classical SVM. Step 1 does an initialization of the root filter by training an SVM on the non-vehicle negative samples. Step 2 forces the filter weights to take the peculiar shape of vehicle class c by data mining on a subset of vehicle negative samples. So initializing parts position after this step gives a better discriminative set of parts. Figure 2 (first row) illustrates the root filter weights after steps 1 and 2. Step 3 is discussed in the next section. Steps 4 and 5 train the whole model using latent SVM and data mining on vehicle negative examples.

While training a model, we often use a large and unbalanced training dataset. In the situation when the number of negative examples are much larger than the number of positive examples, it is infeasible to consider all the negative examples simultaneously. So we used a multi-class version of the bootstrapping-like approach proposed in [4] for all training steps. Early steps just use a subset of all negative examples uniformly for data-mining but with a larger number of iterations in comparison to the final step which uses all the negative examples. For example, CompCars dataset has 281 classes, each of which has 150 samples on average. Hence, by using a classical SVM, training is done on 150 positives and $281 * 150 = 42$k negative examples. This is an extremely imbalanced binary classification problem and would probably lead to overfitting even with a weighted SVM. We reduced the classifier generalization error by employing just a small fraction of all negative samples (i.e. 30 samples per negative class: $281 * 30 = 8$k). We call this subset as the K-uniform subset of negative samples. In order to evaluate this new

Procedure Train

Function M_c = **Train**(Model M_c, posSet, negSet, n, alg)
  $M_{0..n} = \{M_c \dots M_c\}$
  $loss_0 = \infty$
  for i = 1:n
    negs: data mining hard negatives by M_c
    M_i: Train M_c with {posSet, negs} using alg procedure
    $loss_{dm}$ = compute hinge loss on negs
    if $(loss_{i-1} - loss_i) < \varepsilon$
      $M_n = M_i$
      break
    end
    $loss_{i-1} = loss_i$
  end
  $M_c = M_n$
End

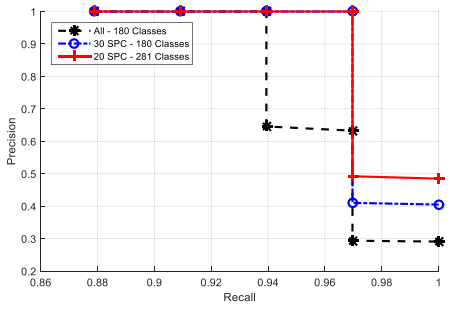

Fig. 3. Precision/Recall curves for a model trained on all negative images of 180 classes (All), a model trained on 30 SPC of 180 classes and a model trained on 20 SPC of 281 classes. Training is done on 'Besturn x80 366' class of CompCars dataset.

scheme, an experiment has been performed. A model is trained on a specific class with three different data-mining configurations. First model is trained with all negative images of just 180 classes of CompCars dataset. Second model is trained with only 30 SPC of 180 classes and the third model is trained with 20 SPC of 281 classes. Figure 3 illustrates precision/recall curves of these three models. The 30 SPC version obtained better precision with a smaller number of training examples on CompCars test set. Although the 20 SPC version is used on the harder training problem with 281 classes, it gained higher precision/recall.

M_c is the model we want to retrain. $posSet$ and $negSet$ are positive and negative sets that the training should be done on. n specifies the number of training iterations. Each iteration consists of a data mining step and a training step. The Train procedure has two termination conditions: 1) reaching the maximum number of iterations, or 2) small reduction (ε) of current model's error in comparison to the model in the previous iteration.

B. Finding Discriminative Parts Set

We proposed an algorithm to find a discriminative set of parts for a specific vehicle make and model. An energy-based greedy algorithm was used in [4] for finding the best set of parts. This algorithm iterates through the currently trained root filter weights and greedily selects n rectangular regions

Function P = **InitializePartsLocation**(F_{rc}, F_{ro})
  n : number of parts, S: fixed size of parts
  P : ∅
  for i = 1:n
    sims :$[\infty]_{1\times|F_{rc}|}$
    for r_c of size S ∈ F_{rc}
      sims[r_c] = Sim(r_c, F_{ro})
    end
    r_c = find r so that $\min_r sims[r]$
    P = P ∪ r_c
  end
end

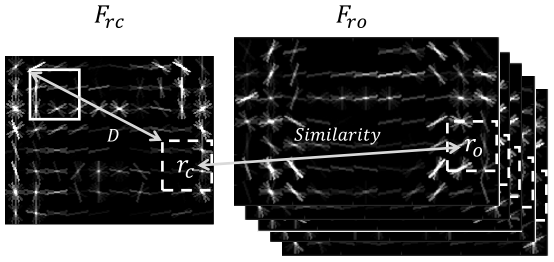

Fig. 4. The proposed greedy function tries to maximize D and minimize Similarity [3].

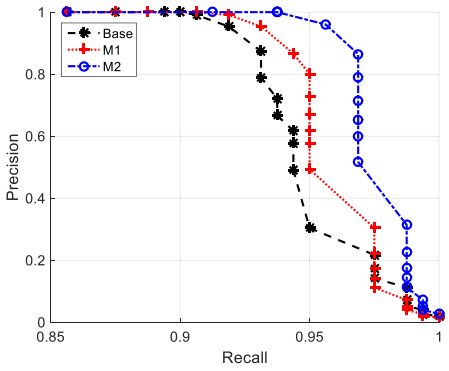

Fig. 5. Precision/Recall curves for a model trained by the baseline algorithm [4] (Base), after applying the proposed Learn procedure (M1) and after utilizing the proposed parts localization method (M2). Training is done on 'Pride 141' class in BVMMR dataset [3].

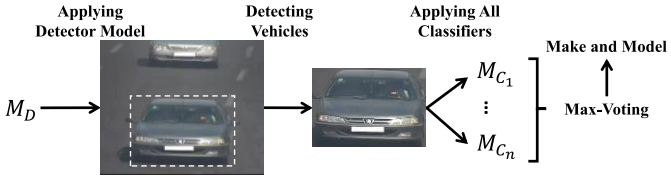

Fig. 6. The steps used by the proposed system to process an input image.

of size S that has the most positive energy. After selecting a part, they zero out the weight in that region of the root filter and repeat the operation. This algorithm works for a binary classification problem. However, it is not adequate for a fine-grained recognition problem which has a large number of classes. Our algorithm considers the root filters of all classes to find discriminative parts set for class c. In our definition, a part belongs to class c is called discriminative if its similarity to the equivalent part (in terms of location and size) in other classes is low. Assume we have the root filters of M classes ($F_r = \{F_{rc_1}, F_{rc_2}, \ldots, F_{rc_M}\}$) and we want to find n discriminative parts of class c. In the following, F_{rc} is the root filter of class c and F_{ro} is the set of other classes root filter:

$$F_{rc} = F_{rc_i}, \quad F_{ro} = F_r \backslash F_{rc_i}$$

The similarity of part r_c for class c in a specific location of F_{rc} (l_r) to the equivalent part in F_{ro} is calculated as follows:

$$Sim(r_c, F_{ro}) = \frac{1}{|F_{ro}|} \sum_1^{|F_{ro}|} cos(r_c, r_o) + \alpha \cdot (1 - D(l_r, l_i)) \tag{3}$$

Function Sim iterates through all parts r_o in F_{ro} with the same location and size as r_c and calculates the similarity between them. $cos\,(r_c, r_o)$ returns the cosine measure between part r_c and r_o. D is the normalized Euclidean distance between part r_c location (l_r) and the nearest previously selected part location of class c to part r_c (l_i). α is the regularization constant. $\alpha > 0.5$ assigns an acceptable penalty to overlap of the selected parts (Figure 4). The proposed algorithm using this similarity measure is presented below.

A region is a part candidate if its distance measure ($-1 * similarity$) is the highest in comparison to the same region in other classes root filter. Employing this logic, procedure InitializePartsLocation finds the n first best part candidates. The effectiveness of this algorithm is improved by putting it in Step 3 of the Learn procedure. To better observe the effect of the proposed part localization, the following experiment is conducted. A single classifier is trained using three different approaches: 1) the baseline algorithm presented in [4] (Base), 2) after applying the proposed Learn procedure without parts localization (M1), and 3) after applying the Learn procedure with the proposed parts localization (M2). Figure 5 depicts precision/recall curves for these three methods on 'Pride 141' class of BVMMR dataset. Furthermore, Figure 2 (second row) shows a fully-trained model for "Besturn x80 366" class of CompCars dataset by 8 parts.

C. Cascading Scheme

In order to be able to detect vehicles in the preprocessing stage, we trained a simple and fast car detector M_d using HOG and SVM like in [36]. After detecting vehicle bounding boxes in the input image, all models $\{M_{C_i}, \ldots M_{C_n}\}$ are applied to each vehicle in the image simultaneously. The model with the maximum score decides the output. If all model scores were less than a small threshold, the vehicle subcategory would be reported as "Unknown". The steps used by the proposed system to process an input image is illustrated in Figure 6.

In a fine-grained recognition problem, there are often many classes or subcategories. There are a number of algorithms for combining individual binary classifiers, but almost all of them apply all of the classifiers to the input image, like we did in our baseline scheme shown in Figure 6. Applying a large number of classifiers to the input image, even in a parallel manner is a time-consuming process. The proposed cascading scheme decreases the system overall processing time substantially

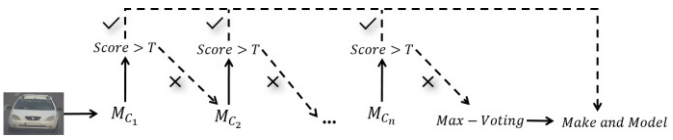

Fig. 7. The steps used by the proposed cascading scheme to process an input image.

with employing just a subset of all classifiers for each input image. Our cascading scheme includes a sequential ordering of models $\{M_{C_i}, \ldots M_{C_n}\}$ and a global threshold T. There is one model in each stage of the cascade. The input image is given to the first model in the cascade, if the model accepts the image (the model's score is greater than T), no further processing is done. Otherwise, the image moves to the next stage. If none of the models accept the image, the max-voting scheme will be used. Figure 7 demonstrates the proposed cascading scheme.

We arrange the models in the order which results in a higher speed and analogous accuracy in comparison to the baseline system. Two properties of a classifier for attaining an effective and practical ordering, namely confidence and frequency are proposed. One can gain high accuracy by putting a model with the maximum confidence in the first stage of the cascade. In our definition, a model is the most confident, if its entropy is the lowest. Such a model accepts a positive input with a high score and rejects a negative input with a low score. In the other hand, high speed can be achieved by putting the most probable model in the first stage of the cascade. We can estimate the prior probability of an input image belonging to a specific class, by considering the distribution of each class in the training set. The following formula offers our proposed combination of the two aforementioned factors:

$$C_i = \alpha E_i + (1-\alpha) F_i \tag{4}$$

$$E_i = -\frac{1}{\sum_{j=1}^{K} M_i\left(x_j\right) \log(M_i\left(x_j\right))} \tag{5}$$

K is the number of training samples and $M_i(x_j)$ is the rescaled and normalized score of model M_i for sample X_j in range of $[0-1]$. The model M_i with greater C_i will be placed nearer to the first stage of the cascade. E_i and F_i are the normalized inverse of entropy and frequency of model M_i respectively. Constant $0 \leq \alpha \leq 1$ controls the tradeoff between E_i and F_i. Different values of α yield to different orderings of models in the cascade.

Cascading the system requires selection of the proposed cascade parameters, α and the global threshold T. The parameters selection is done empirically. In the following experiment, the accuracy and speed of the cascaded system have been measured for different values of α and T within the range of $[0-1]$ and $[0.5-0.8]$ respectively on the BVMMR training set. Figure 8 and Figure 9 illustrate the cascading scheme effect on system accuracy and speed.

It can be seen that the selection of $\alpha = 0.8$ and $T = 0.7$ results in a 30% speedup and a small improvement in accuracy. The global threshold establishes a balance between E_i and F_i. A small global threshold grows the dependency to the ordering

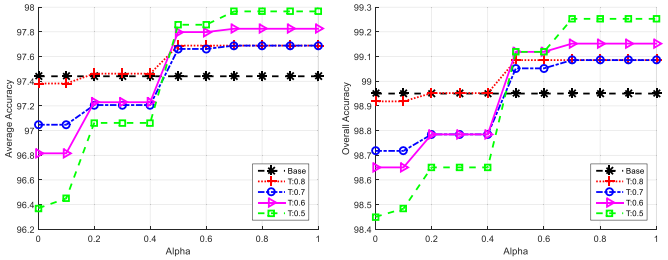

Fig. 8. The cascaded system average and overall accuracy based on different values of $\boldsymbol{\alpha}$ and $\boldsymbol{T}$. Base refers to the non-cascaded system.

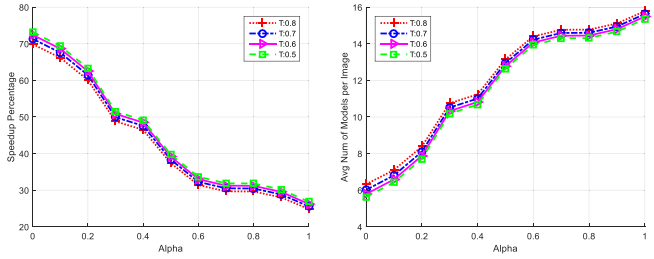

Fig. 9. The cascaded system speedup percentage and average number of applied models per image based on different values of $\boldsymbol{\alpha}$ and $\mathbf{T}$.

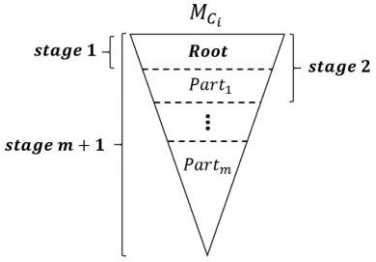

Fig. 10. Structure of an individual cascade model.

of the models in the cascade and a large global threshold decreases the cascade speed. Accordingly, a reasonable global threshold is $T = 0.7$. Calculation of α and T is done only once on a training set. As in our experiments, we selected these two criteria based on the results presented in Figure 8 and Figure 9 on BVMMR training set, and we use the same values on the CompCars dataset.

Currently, as shown in the two aforementioned figures, our cascading scheme can achieve up to 70 percent speedup with having the same accuracy as the baseline (non-cascaded) system on BVMMR training set. But if a low processing time is crucial, we can sacrifice a bit of accuracy for a higher speed by replacing the classifiers with their cascaded versions [37]. In a typical model with one root and eight parts, nine filters are applied to each possible location and scale of the input image. We put the root filter in the first stage of the cascade, so other parts only apply to regions which have passed the first stage. Each stage S_i has a threshold T_i. These thresholds are learned from training samples. While applying cascade models to the input image, a model M_j will be dropped if its current stage score $Score_{M_j} < T_{i_{M_j}}$. $T_{i_{M_j}}$ is the threshold of stage i of model M_j. Figure 10 depicts the structure of an individual cascade model.

Fig. 11. Some cropped sample images from BVMMR dataset.

Fig. 12. Some sample images from CompCars surveillance dataset.

For measuring the speedup of this modification, an experiment is done on BVMMR and CompCars datasets, by replacing all models in the baseline system with the cascade models. The results indicate that the system didn't use nearly 80% of the overall number of filters from all 21 models ($21 \times 9 = 189$) for an input image of BVMMR dataset on average. For CompCars dataset, this percentage is around 74% of the overall number of filters from all 281 models ($281 \times 9 = 2529$). This means at least, more than 50% speedup, assuming root score takes longer than part scores to compute.

D. Models Calibration

Since the models are trained individually and not calibrated, their thresholds and output range are not comparable. Platt-Scaling is performed for calibrating the models by fitting a probability distribution over the training samples. The calibrated scores of a model's output $f(x)$ are as follows:

$$P\,(y = 1|\,x) = \frac{1}{1 + e^{Af(x)+B}} \tag{6}$$

The parameters A and B are estimated using the training examples. Furthermore, Platt has proposed to transform labels 1 for positive and -1 for negative samples to the following probabilities:

$$y_+ = \frac{N_+ + 1}{N_+ + 2} \quad y_- = \frac{1}{N_- + 2}$$

Constants N_+ and N_- are the number of positive and negative samples respectively. While the logistic fitting is performed independently for each model, we found that it gives us a considerable boost in overall recognition performance over using raw model's scores.

TABLE I

THE PROPOSED SYSTEM CLASSIFICATION ACCURACY (%) OF EACH CLASS AT RANK 1, RANK 3 AND RANK 5 ON BVMMR DATASET

Class Index	Rank 1	Rank 3	Rank 5	Class Index	Rank 1	Rank 3	Rank 5
1	88.89	88.89	88.89	12	99.90	100	100
2	94.74	94.74	100	13	99.20	99.20	99.20
3	100	100	100	14	96.25	97.50	97.50
4	100	100	100	15	100	100	100
5	94.74	100	100	16	94.74	100	100
6	91.54	100	100	17	100	100	100
7	100	100	100	18	99.59	100	100
8	100	100	100	19	91.66	95.83	100
9	96.83	100	100	20	100	100	100
10	93.08	100	100	21	88.89	100	100
11	100	100	100	22	100	100	100

IV. EXPERIMENTAL RESULTS

A. The Datasets

Although there are several works published on VMMR, there was no standard benchmark dataset for vehicle makes and models until recently. Most of the related works reported their results on their own dataset. Unlike most the prior works, we shared our dataset BVMMR dataset v2 including test-train partitioning and developmental kit*.

The BVMMR dataset includes 4858 images with 5991 cars from 28 different makes and models. Images are in various lightning conditions and different resolutions. Figure 11 shows some cropped sample images from this dataset. The images in the dataset are annotated according to the pascal standard [38].

The BVMMR dataset is splitted into two almost equals partitions for training and testing. No annotation has been used in the testing phase and the cars have been detected using our car-detector with 100% precision. In experiments, 21 classes of vehicles with more than 20 samples have been selected and the others are added to "Unknown" class. The "Unknown" class is added to this set to increase the difficulty of the classification task. Any vehicle which has not been classified into one of the 21 subcategories, goes to "Unknown" subcategory.

Recently, two datasets have been provided [7], [39] for VMMR. NTOU-MMR is the first one which doesn't have any annotation and can't be used in the experiments. CompCars is the second one which consists of a web-nature part, made of 136k of vehicles from nearly 1600 classes of vehicles taken from different viewpoints, and a surveillance-nature part with 50k frontal images of 281 classes of vehicles taken from surveillance cameras. Our system has a restriction on vehicles viewpoints that indicates all images should have a similar view. Hence, we used the surveillance images of CompCars in our experiments. These images have diverse lighting conditions and very similar makes and models. Figure 12 demonstrates some sample images from CompCars surveillance dataset.

B. Evaluation of the Proposed System

In the entire system, HOG [36] is used for feature extraction. We used the cell size of 8×8 pixels and block size of 2×2

*http://mbt925.ir/publications.html

TABLE II

THE PROPOSED SYSTEM OVERALL/AVERAGE ACCURACY (%) AT RANK 1, RANK 3 AND RANK 5 ON BVMMR DATASET

Rank	1	3	5
Overall Accuracy	98.66	99.73	99.86
Average Accuracy	96.82	98.41	99.34

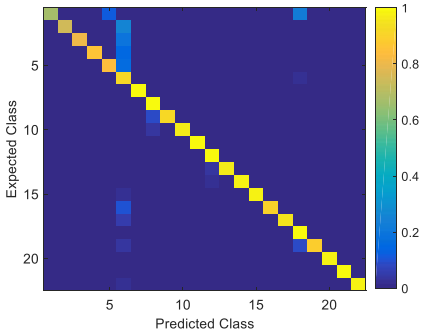

Fig. 13. Confusion matrix of the proposed system at rank 1 on BVMMR dataset [3].

TABLE III

THE PROPOSED SYSTEM OVERALL/AVERAGE ACCURACY (%) AT RANK 1, RANK 3 AND RANK 5 ON COMPCARS DATASET

Rank	1	3	5
Overall Accuracy	97.43	98.41	98.68
Average Accuracy	95.50	97.27	97.78

empirically. The proposed system doesn't have a constraint on a specific feature extraction algorithm. We picked HOG because of its speed and simplicity, so we could focus on the training algorithm structure. Other feature extraction methods can be employed.

Table I provides the recognition accuracy of the proposed baseline system for each class of BVMMR dataset at rank 1, rank 3 and rank 5. Table II gives the overall and average accuracies calculated from Table I. Confusion matrix at rank 1 is presented in Figure 13. The overall and average accuracies are measured as follows:

$$Overall\ Accuracy = \frac{p}{N} \tag{7}$$

$$Average\ Accuracy = \frac{\sum_{i=1}^{N} \frac{p_i}{n_i}}{N} \tag{8}$$

p represents the total number of correctly classified test samples and N represents the total number of test samples. p_i is the number of correct predictions on i-th class and n_i is the number of samples of i-th class.

Considering the variations in the resolution and appearances of BVMMR dataset images, the reported accuracies are acceptable. Our system gained a high classification rate even for classes with small number of training examples, i.e. classes 1 to 5. Using an appropriate threshold for the trained models, the system correctly classified over 91% of the images belonging to "Unknown" classes at rank 1. Classes with a large number of examples led to more powerful models, as we can observe the models trained for classes 8, 12 and 18 gained high performances at rank 1 with small changes across rank 3 and

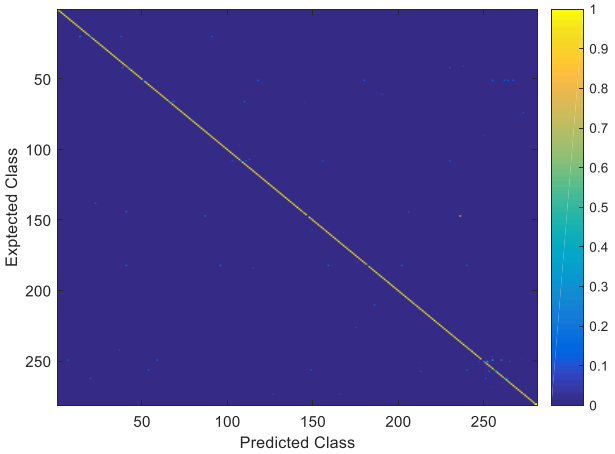

Fig. 14. Confusion matrix of the proposed system at rank 1 on CompCars dataset.

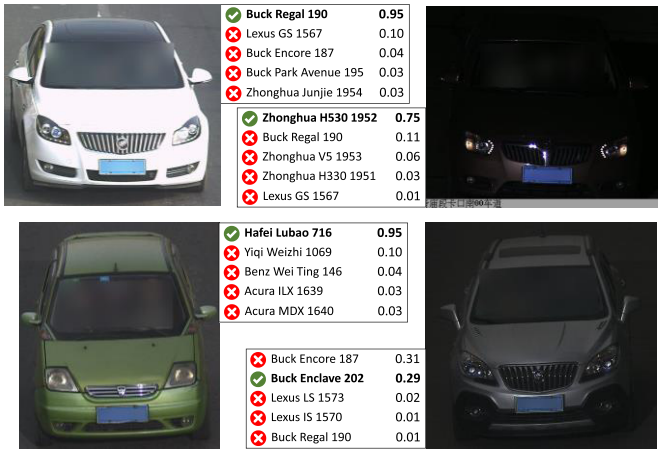

Fig. 15. The proposed system top 5 predicted classes for a few input samples of CompCars dataset.

rank 5. The confusion matrix in Figure 13 demonstrates a high accuracy among all classes and a few misclassified cases.

The total and average accuracy of our system on CompCars dataset are given in Table III. Due to the large number of classes of CompCars dataset, presentation of individual class recognition rates was not possible. However, the displayed confusion matrix at rank 1 in Figure 14 gives an overview of the accuracy and misclassification rate of each class.

The surveillance images of CompCars dataset have very small inter-class distance. Also, there are some images from each make and model which were taken at different lighting condition (i.e. day, night, sunny and foggy). Taking these descriptions into account, our system performance was notable. Figure 15 and Figure 16 display a few examples of our system's output on CompCars and BVMMR datasets.

The proposed system average processing times on all images of BVMMR and CompCars datasets on a 2.50GHz Intel Corei7-4710HQ CPU computer are presented in Table IV and Table V. All the reported accuracies are at rank 1. Parallel versions of the system simply apply the models to the input image simultaneously.

Our computer processor has 4 cores, therefore the decreases from 11 to 3.1 seconds in Table IV and 420 to 120 seconds in Table V are quite good, considering parallelization

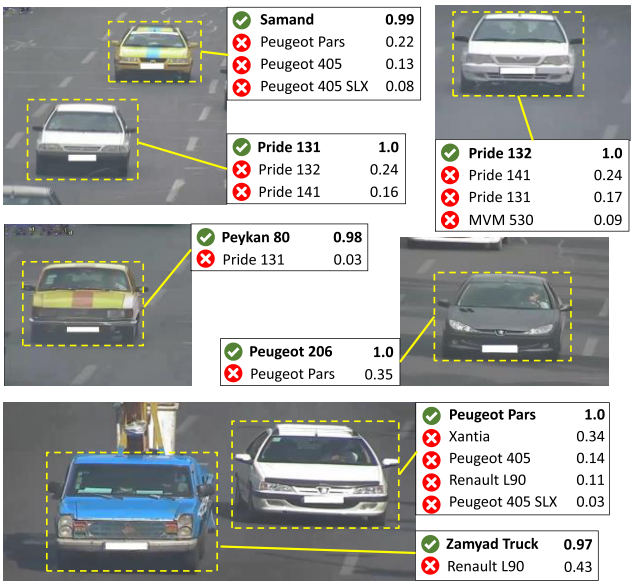

Fig. 16. The proposed system top 5 predicted classes for a few input samples of BVMMR dataset. Some samples have less than 5 predicted classes (There were no other model with score > 0 for them).

TABLE IV

AVERAGE PROCESSING TIME AND ACCURACY (%) OF DIFFERENT CONFIGURATIONS OF THE PROPOSED SYSTEM ON BVMMR DATASET

Configuration	α	Time (s)	Average Accuracy	Overall Accuracy
1. Baseline system – serial	-	11	96.82	98.66
2. Baseline system – parallel	-	3.1	96.82	98.66
3. Cascade of models – serial	0.8	5.8	97.01	98.80
4. Cascade of models – serial	0	1.7	96.42	98.66
5. Cascade of models – parallel	**0.8**	**2.6**	**97.01**	**98.80**
6. Cascade of models – parallel	0	0.8	96.42	98.66
7. Cascade of cascade models – serial	0.8	3.5	92.15	97.83
8. Cascade of cascade models – serial	0	0.8	91.46	97.70
9. Cascade of cascade models – parallel	0.8	1.2	92.15	97.83
10. Cascade of cascade models – parallel	**0**	**0.2**	**91.46**	**97.70**

overheads. The best accuracy is achieved by the configuration number 5 of the system on both datasets. This configuration does not decrease the recognition rate at a speed faster than even the parallel version of the non-cascaded system. The last configuration results in an about 90% speedup on BVMMR and 70% speedup on CompCars dataset with just about 1% drop in overall accuracy. The gap between average accuracy of configurations number 5 and 10 is caused by improper calculation of intermediate thresholds of the cascade classifiers for subcategories with a small number of training examples. Besides, one misclassified sample in these categories yields a major reduction in average accuracy. It is worth to note that the dimension of images in CompCars are 1.5x larger than the images in BVMMR dataset.

C. Comparison With Other Methods

To the best of our knowledge, a few works are available on surveillance images of CompCars dataset up until now [33].

TABLE V

AVERAGE PROCESSING TIME AND ACCURACY (%) OF DIFFERENT CONFIGURATIONS OF THE PROPOSED SYSTEM ON COMPCARS DATASET

Configuration	α	Time (s)	Average Accuracy	Overall Accuracy
1. Baseline system – serial	-	420	95.50	97.43
2. Baseline system – parallel	-	120	95.50	97.43
3. Cascade of models – serial	0.8	230	95.55	97.52
4. Cascade of models – serial	0	210	95.32	97.27
5. Cascade of models – parallel	**0.8**	**88**	**95.55**	**97.52**
6. Cascade of models – parallel	0	75	95.32	97.27
7. Cascade of cascade models – serial	0.8	142	94.84	96.77
8. Cascade of cascade models – serial	0	128	94.39	96.61
9. Cascade of cascade models – parallel	0.8	47	94.84	96.77
10. Cascade of cascade models – parallel	**0**	**33**	94.39	96.61

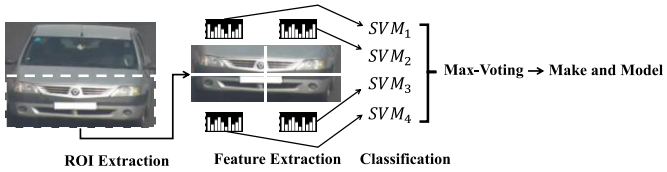

Fig. 17. The implemented multi-region approach with 2x2 grid and one classifier per cell.

So in order to do a more thorough comparison, we implemented a multi-region ROI-based approach commonly used by the previous works. For ROI extraction in this approach, the input image is divided vertically and the bottom half is selected as the ROI. This spatial pyramid divides the ROI to a $N \times M$ grid, extracts features from each cell separately and trains a classifier per cell (Figure 17). Different grid sizes are tested. In order to have a fair comparison, HOG is employed with the same parameters as before. SVM classifier is used with one-against-all scheme for classification by LIBSVM library [40]. BVMMR dataset is imbalanced, so we used weighted SVM (C-SVM algorithm in LIBSVM library) with larger weights for classes with smaller number of samples, such that the weight $w_i = \frac{N}{N_i}$ is assigned to class i. N_i is the number of training examples of class i, and N is the total number of training examples. In this way, we typically raise the misclassification penalty of small classes. In the other hand, standard SVM is utilized for CompCars dataset.

Table VI and Table VII present the comparison results at rank 1 for BVMMR and CompCars datasets. The "Unknown" class of BVMMR dataset is omitted from this experiment. MULTI_SVM_N_M in the following tables points to multiple SVMs each of which applies to a cell of a N × M grid, while SINGLE_SVM operates on a 1 × 1 grid with a single SVM. As stated before, our system uses just 30 samples of each negative classes (SPC) from CompCars dataset for data mining in the training process. Therefore, a similar approach is employed in the implemented multi-region methods too. Different number of SPCs are tested and the results are reported in Table VII.

TABLE VI

THE PERFORMANCE COMPARISON OF THE PROPOSED SYSTEM WITH MULTI-REGION APPROACHES ON BVMMR DATASET

Approach	Average Accuracy	Overall Accuracy
SINGLE_SVM_1_1	91.28%	93.77%
MULTI_SVM_2_2	89.0%	92.48%
MULTI_SVM_2_3	88.08%	89.26%
MULTI_SVM_2_4	88.20%	89.71%
Proposed Approach (conf. num. 5)	97.07%	99.05%

TABLE VII

THE PERFORMANCE COMPARISON OF THE PROPOSED SYSTEM WITH OTHER APPROACHES ON COMPCARS DATASET

Approach	Average Accuracy	Overall Accuracy
SINGLE_SVM_1_1 (20 SPC)	78.24%	79.81%
SINGLE_SVM_1_1 (25 SPC)	80.15%	81.05%
SINGLE_SVM_1_1 (30 SPC)	82.13%	83.48%
MULTI_SVM_2_2 (30 SPC)	81.43%	79.76%
MULTI_SVM_2_3 (30 SPC)	81.55%	79.82%
MULTI_SVM_2_4 (30 SPC)	81.68%	80.09%
*Hsieh et al. [41]	51.70%	-
*Zhang [42]	83.78%	-
Fang et al. (whole image) [33]	90.78%	-
Fang et al. (whole image + parts)	98.29%	98.63%
Proposed Approach (conf. num. 5)	95.55%	97.52%

* These methods are implemented by [33].

On BVMMR datasets, our system outperforms other methods. However, the approach of [33] performed better than ours on CompCars dataset. Their better accuracy is mainly due to the CNN feature extraction power. Likewise, our approach will probably performs better by using CNN features instead of HOG. Besides, they used 4 CNNs and multiple SVMs together which need more computational power, memory and time than the proposed system.

Additionally, Single_SVM performed better than MULTI_SVM. This is probably due to the partitioning scheme that leads to a poor regions selection. As the problem gets harder with the growing number of classes from Table VII to Table VII, the difference amplifies between the accuracies of the proposed approach and multi-region approach.

V. CONCLUSION

In this paper, we proposed a part-based approach for vehicle make and model recognition, together with a novel cascading scheme. Our approach finds a discriminate set of parts for each vehicle subcategory using an iterative training algorithm and a novel discriminative part localization. Our system learns a model for each class of vehicles, while simultaneously searches for discriminative parts of that class. Using a novel multi-class data mining algorithm, the training procedure can be done optimally in terms of memory and time, and yet produce a powerful model. The performance improvements gained from each of these proposed parts are demonstrated empirically. Furthermore, a novel cascading scheme is proposed to enhance the speed of the system. One of the desirable configuration of our cascaded system performs with higher speed and accuracy than the baseline system. Furthermore, the fastest version of the cascaded system can process an input image up to 80% faster with only a minor sacrifice in overall accuracy.

We were able to achieve an average accuracy of 97.01% on our challenging dataset. In addition, our system is evaluated on the recently published comprehensive CompCars dataset, and achieved a formidable average accuracy of 95.55%.

REFERENCES

[1] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 694–711, May 2006.

[2] K. Yousaf, A. Iftikhar, and A. Javed, "Comparative analysis of automatic vehicle classification techniques: A survey," *Int. J. Image, Graph. Signal Process.*, vol. 4, no. 9, pp. 52–59, 2012.

[3] M. Biglari, A. Soleimani, and H. Hassanpour, "Part-based recognition of vehicle make and model," *IET Image Process.*, vol. 11, no. 7, pp. 483–491, Jul. 2017, doi: 10.1049/iet-ipr.2016.0969.

[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[5] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2002, pp. 577–584.

[6] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Adv. Large Margin Classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

[7] 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.

[8] M. Conos, "Recognition of vehicle make from a frontal view," M.S. thesis, Dept. Math., Czech Tech. Univ. Prague, Prague, Czech Republic, 2007.

[9] X. Clady, P. Negri, M. Milgram, and R. Poulenard, "Multi-class vehicle type recognition system," in *Artificial Neural Networks in Pattern Recognition* (Lecture Notes in Computer Science). Springer, 2008, pp. 228–239.

[10] G. Pearce and N. Pears, "Automatic make and model recognition from frontal images of cars," in *Proc. 8th IEEE Int. Conf. Adv. Video Signal Based Surveill.*, Aug. 2011, pp. 373–378.

[11] S. Saravi and E. A. Edirisinghe, "Vehicle make and model recognition in CCTV footage," in *Proc. 18th Int. Conf. Digit. Signal Process.*, Jul. 2013, pp. 1–6.

[12] D. T. Munroe and M. G. Madden, "Multi-class and single-class classification approaches to vehicle model recognition from images," in *Proc. 16th Irish Conf. Artif. Intell. Cognit. Sci.*, 2005, pp. 93–102.

[13] N. Boonsim and S. Prakoonwit, "Car make and model recognition under limited lighting conditions at night," *Pattern Anal. Appl.*, 2016, doi: https://doi.org/10.1007/s10044-016-0559-6

[14] V. S. Petrović and T. F. Cootes, "Analysis of features for rigid structure vehicle type recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2004, pp. 587–596.

[15] Y. Yuan, Z. Xiong, and Q. Wang, "An incremental framework for video-based traffic sign detection, tracking, and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 7, pp. 1918–1929, Jul. 2017.

[16] J.-W. Hsieh, L.-C. Chen, D.-Y. Chen, and S.-C. Cheng, "Vehicle make and model recognition using symmetrical SURF," in *Proc. Adv. Video Signal Based Surveill.*, Aug. 2013, pp. 472–477.

[17] 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, 2011.

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

[19] D. Santos and P. L. Correia, "Car recognition based on back lights and rear view features," in *Proc. 10th Int. Workshop Image Anal. Multimedia Interact. Services*, May 2009, pp. 137–140.

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

[21] M. S. Sarfraz and M. H. Khan, "A probabilistic framework for patch based vehicle type recognition," in *Proc. VISAPP*, 2011, pp. 358–363.
[22] X. Li, M. Chen, and Q. Wang, "Measuring collectiveness via refined topological similarity," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 12, no. 2, pp. 34:1–34:22, 2016.
[23] Q. Wang, J. Fang, and Y. Yuan, "Adaptive road detection via context-aware label transfer," *Neurocomputing*, vol. 158, pp. 174–183, Jun. 2015.
[24] J. Sochor, A. Herout, and J. Havel, "BoxCars: 3D boxes as CNN input for improved fine-grained vehicle recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3006–3015.
[25] 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.
[26] Y. Gao and H. J. Lee, "Vehicle make recognition based on convolutional neural network," in *Proc. 2nd Int. Conf. Inf. Sci. Secur.*, Dec. 2015, pp. 1–4.
[27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
[28] Y. Zhou, L. Liu, L. Shao, and M. Mellor. (2016). "DAVE: A unified framework for fast vehicle detection and annotation," pp. 1–16. [Online]. Available: https://arxiv.org/abs/1607.04564
[29] R. Girshick, F. Iandola, T. Darrell, and J. Malik, "Deformable part models are convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 437–446.
[30] H. Huttunen, F. S. Yancheshmeh, and K. Chen. (2016). "Car type recognition with deep neural networks." [Online]. Available: https://arxiv.org/abs/1602.07125
[31] M. Liu, C. Yu, H. Ling, and J. Lei, "Hierarchical joint CNN-based models for fine-grained cars recognition," in *Proc. Int. Conf. Cloud Comput. Secur.*, 2016, pp. 337–347.
[32] Y. Gao and H. J. Lee, "Local tiled deep networks for recognition of vehicle make and model," *Sensors*, vol. 16, no. 2, p. 226, 2016.
[33] J. Fang, Y. Zhou, Y. Yu, and S. Du, "Fine-grained vehicle model recognition using a coarse-to-fine convolutional neural network architecture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 7, pp. 1782–1792, Jul. 2016.
[34] M. S. Sarfraz, A. Saeed, M. H. Khan, and Z. Riaz, "Bayesian prior models for vehicle make and model recognition," in *Proc. 6th Int. Conf. Frontiers Inf. Technol.*, 2009, Art. no. 35.
[35] K. Duan, L. Marchesotti, and D. J. Crandall, "Attribute-based vehicle recognition using viewpoint-aware multiple instance SVMs," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2014, pp. 333–338.
[36] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 886–893.
[37] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2241–2248.
[38] (2008). *The PASCAL Visual Object Classes*. Accessed: Mar. 10, 2015. [Online]. Available: http://pascallin.ecs.soton.ac.uk/challenges/VOC/
[39] *NTOU-MMR Dataset*. Accessed: Oct. 22, 2016. [Online]. Available: http://mmplab.cs.ntou.edu.tw/mmplab/MMR/MMR.html
[40] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, Apr. 2011, Art. no. 27.
[41] 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. Pattern Anal. Mach. Intell.*, vol. 15, no. 1, pp. 6–20, Feb. 2014.
[42] B. Zhang, "Reliable classification of vehicle types based on cascade classifier ensembles," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 1, pp. 322–332, Mar. 2013.

Mohsen Biglari received the B.Sc. and M.Sc. degrees in software engineering from the School of Electronic and Computer Engineering, Kashan, Esfahan, Iran, in 2009 and 2011, respectively. He is currently pursuing the Ph.D. degree with the Image Processing and Data Mining Laboratory, Shahrood University of Technology, Semnan, Iran. His current research interests include machine learning, image processing, and parallel computing.

Ali Soleimani received the degree in electrical engineering and the M.Sc. degree in 1994, and the Ph.D. degree in electrical engineering from the Iran University of Science and Technology, Tehran, Iran, in 2000. As a master student, he started his research on fuzzy logic and applications, nonlinear MIMO systems, and controlling them by neuro-fuzzy controller. Since 2001, he has been a Professor of digital electronics with the Shahrood University of Technology, Iran. His current research interests include digital signal processing, neural network, fuzzy logic, and FPGA hardware and software.

Hamid Hassanpour received the B.S. degree in computer engineering from the Iran University of Science and Technology, Tehran, Iran, in 1993, the M.S. degree in computer engineering from the Amirkabir University of Technology, Tehran, in 1996, and the Ph.D. degree from the Queensland University of Technology, Brisbane, Australia, in 2004. He is currently a Professor with the Faculty of Computer Engineering and IT, Shahrood University of Technology, Iran. His current research interests include image processing, signal processing, and time frequency signal processing and analysis.