Journal of Computers Vol. 31 No. 6, 2020, pp. 24-37
doi:10.3966/199115992020123106003

Approaches of Transfer Learning and Fine-Tuning on the Effects of Performance of Vehicle Classification

Deng-Yuan Huang[1*], Wen-Lung Lin[2], Yu-Yun Wang[3]

[1] Department of Computer Science and Information Engineering, Da-Yeh University, Changhua 515, Taiwan
kevin@mail.dyu.edu.tw

[2,3] Department of Electrical Engineering, Da-Yeh University, Changhua 515, Taiwan
{dragon830816, sophia880122}@gmail.com

Received 24 July 2019; Revised 17 January 2020; Accepted 2 March 2020

Abstract. In this paper, we carry out comparisons of transfer learning and fine-tuning methods on the performance of vehicle classification in a relatively small dataset. These two methods are used for the alleviation of the overfitting problem and the reduction of the data labeling costs. For deep learning-based classification task, sufficient training data is very important, but sometimes the collection of training data is quite difficult, especially for some domain knowledge fields, e.g., medical images. Therefore, the investigation of deep learning-based classification problem for a relatively small dataset is still valuable. Transfer learning is a method that uses a pre-trained deep convolutional (CONV) neural network to learn patterns from data that are not seen before, and it is often served as a feature extractor. Fine-tuning can be considered as another type of transfer learning, but its performance is usually better than transfer learning, provided there has sufficient training data. Experimental results show that for transfer learning, the average recall rates are all the same to be 93% when either the classifier of the linear SVM or Logistic regression is applied on the top of the network architecture. But for fine-tuning, the average recall rate can be further increased to 95%, indicating that fine-tuning outperforms transfer learning to the task of vehicle classification.

Keywords: deep learning, fine-tuning, regression, support vector machine (SVM), transfer learning

1 Introduction

With the advent of deep learning, artificial intelligence (AI) has now entered a new era to thrive and flourish. Besides deep learning, big data and large-scale parallel computation are also considered as the main forces to drive the advancement of AI. According to the definition of deep learning by Wikipedia [1], deep learning is a branch of machine learning to extracting features from massive dataset through multilayer nonlinear transformation. Since deep neural networks (DNNs) are the most commonly used method for implementing multilayer nonlinear transformation, deep learning is also regarded as to be synonymous with DNN in practical applications. According to the definition given in Wikipedia earlier, deep learning has two very important characteristics, namely multilayer and nonlinearity, which allows deep learning to achieve great successes in the field of computer vision [2], including the popular tasks of image classification [3-4], object detection [5-7], semantic segmentation [8-10], style transfer [11-12], image colorization [13-14], image reconstruction [15-16], image super-resolution [17-18] and image synthesis [19-20]. Additionally, common deep learning network architectures include DNN, deep belief network (DBN) [21], convolution neural network (CNN) [3-5, 8-9], recurrent neural network (RNN) [15] and generative adversarial network (GAN) [19] so on.

* Corresponding Author

Almighty artificial intelligence has always been the ultimate goal of humanity and deep learning is the technology that is recognized as the closest to it. In recent years, deep learning has performed quite well in the applications of handwritten recognition [22], image classification [3-4], speech recognition [23], and pedestrian detection [24] so on, all of which are attributed to its deep neural network architecture. Deep learning mimics the multilevel processing capability of the human brain system. Namely, lower-level neurons are used to process low-level information, such as edges; higher-level neurons to handle more abstract meanings, such as shape. Hence, the combination of high- and low-level information forms a more specific image to achieve human cognition to external objects.

In past decades, deep learning has received a lot of attention, mainly from several important contests. In 2009 [25], the winner of handwritten recognition competition was awarded from a deep learning network using the long short-term memory (LSTM) architecture. In 2012 [3], the team directed by Geoffrey Hinton at the University of Toronto in Canada built their deep neural network on the graphic processing units (GPUs) to win the championship in the image recognition contest hosted by the well-known ImageNet Large Scale Visual Recognition Challenge (ILSVRC). In this contest, ILSVRC provided about 1.3 million standardized images with manual annotations. Participants must submit their new ideas of learning rules and then test them with all brand-new images. Results show that the accuracy obtained by the Hinton's group is higher than 10% of traditional methods. In 2015, the team of Microsoft researchers in Beijing won the first place in the ILSVRC competition with an error rate of 4.94% for image recognition, which is lower than 5.1% of the human level. In 2016, Microsoft continued to achieve a word error rate of 5.9% in speech recognition, which is lower than 6.0% of the human level. The most striking thing is that in 2017, AlphaGo developed by the Google DeepMind team beat Jie Ke ranked first in Go with a 3:0 score. These achievements have already marked the beginning of the era when computers have surpassed the human brain.

Deep learning successfully caught a lot of attention in 2012. During this year, Krizhevsky et al. [3] proposed the AlexNet model evaluated on the ImageNet large scale dataset, which is a deep convolutional neural network (DCNN), including five CNN layers and three fully-connected (FC) layers. AlexNet reduced the top-5 error rate to 16.4% on the task of image classification in ILSVRC-2012. But, at the same time, the top-5 error rate was still as high as 26.2% for traditional methods, which was only 2% lower than that achieved in 2011. Astonishing results obtained by AlexNet have greatly inspired researchers all over the world to realize the power of deep learning. As a result, almost all the teams competing in ILSVRC-2013 adopted deep learning-based methods. In this year, the champion on the task of image classification in ILSVRC-2013 was awarded to LeCun's research group, who proposed the so-called ZFNet, from New York University to further reduce the top-5 error rate to 11.7% [26]. In 2014, the top-5 error rate in the ILSVRC contest was constantly reduced to 6.7% by the GoogLeNet that is a 22-layer DCNN proposed by the Google's research team [27]. In 2015, the top-5 error rate was further reduced to the lowest level of 3.6% by the ResNet, which is a 152-layer very deep CNN proposed by He et al. from Microsoft Research Institute [28]. The level of 3.6% error rate has been already lower than 5% erroneously recognized by the human eyes. This result marks the milestone of AI surpassing human perception level.

As mentioned above, the great success of image classification is due to the large image dataset of ImageNet for all the DCNNs, including AlexNet [3], VGGNet [4], ZFNet [26], GoogLeNet [27], and ResNet [28]. However, in many real-world applications, human-annotated training data are quite limited and it inevitably leads to the over-fitting problem in machine learning. In the task of image classification, we often have limited training data in the target domain, but have sufficient training data in another domain of interest. In such cases, if knowledge transfer can be done successfully, the learning performance of classification task can be greatly improved and, thereby, much data labeling effort could be further alleviated.

2 Related Work

Many learning rules only work well under a common assumption, i.e., the gallery and probe dataset is extracted from the same feature space and the same distribution [29]. Once the distribution has been altered, most learning models need to be rebuilt from scratch using newly collected training data. However, the recollection of the new training data and reconstruction of the learning model is much expensive and inconceivable. Therefore, lowering the effort of recollecting the training data becomes

imperative. In such cases, transfer learning between different knowledge domains would be highly desirable.

DCNNs usually require large amounts of training data. For example, the commonly used ImageNet dataset has about 1.3 million images. In the aforementioned DCNNs, such as AlexNet [3], ZFNet [26], GoogLeNet [27], ResNet [28], VGG16 and VGG19 [4], they usually need to spend two to three weeks to train on multiple GPUs. It is computationally intensive and time consuming when training from scratch. Thereby, the strategy of either transfer learning or fine-tuning is imperative and required.

To investigate the mechanism of how features are transferable in deep neural networks, Yosinski et al. [30] conducted several experiments to quantify the generality versus specificity of neurons in each layer of the DCNNs. Several striking results are achieved as follows. (1) The features learned in first top layers appear to be general to any dataset or tasks, but it is specific in the last bottom layers. (2) The transferability of features decreases as the dissimilarity between the base task and target task increases. (3) Initializing a network with transferred features from any number of layers can produce a boost to the generalization that still remains valid even after fine-tuning to the target dataset.

In recent years, transfer learning has been successfully used in many applications, including vehicle classification [31], fire flame detection [32], fish classification [33], flower classification [34], head pose estimation [35], defect classification of printed circuit boards [36], and traffic sign detection and recognition [37]. All of the above tasks are first pre-trained on a large scale dataset in another knowledge domain, and they are then trained on a limited human-annotated training dataset with a specific knowledge domain.

Vehicle classification is an important topic for intelligent transportation systems. Jo et al. [31] proposed a transfer learning-based vehicle classification that is realized using the GoogLeNet [27] pre-trained on the ILSVRC-2012 ImageNet dataset. In their proposed system, Haar-like features were utilized to detect the vehicle area on the roadway video and followed by the transfer learning-based vehicle classification. For the transfer learning, the training parameters in the top layers of the GoogLeNet pre-trained on the ILSVRC-2012 ImageNet dataset are frozen, revealing that the gradient updates in the backward pass of backpropagation can be done only in the bottom layers.

The transfer learning directly migrates the weighted parameters of the DCNN's model to a new task, in which the model has been well trained by the large-scale ImageNet dataset. Fine-tuning can be considered as another type of transfer learning method, but it retrains the weights of some specific convolutional layers that have been well trained on the ImageNet dataset. Therefore, fine-tuning usually takes much more time to tune the weighted parameters than that of transfer learning. Strategies such as transfer learning and fine-tuning usually work well to achieve good results in the classification tasks. Experimental results conducted by Razavian et al. [38] show that it is possible to obtain satisfactory results using the DCNN's model pre-trained by the ImageNet dataset for feature extraction and then use the features extracted to train a new model with different knowledge domains.

In many real-world applications, we often encounter the scenarios to carry out a classification task in one domain of interest with small training data. On the other hand, we have sufficient training data in another domain of interest. The purpose of this work is to overcome the dilemma mentioned above and further to propose the methods such as transfer learning and fine-tuning to alleviate the requirement of large amounts of training data in the DCNN's model because the cost of data labeling is expensive and time-consuming. Furthermore, the comparisons for both transfer learning and fine-tuning in terms of accuracy and time consumption are also presented to provide users better knowledge of these two methods.

The major contributions of this paper can be summarized as follows: (1) DCNNs essentially have large numbers of training parameters. It inevitably causes the overfitting problem when a limited scale dataset is used. The proposed methods such as transfer learning and fine-tuning can effectively alleviate the overfitting problem due to the use of small training dataset. (2) The proposed methods can effectively transfer parameters between DCNN's models with different tasks in various domains of knowledge, by which we can further reduce both requirements of large amounts of training data and the cost of data labeling.

The rest of this paper is organized as follows: Section 3 describes the framework of the proposed method of vehicle classification. The experimental results are given and discussed in Section 4. Finally, the concluding remarks are provided in Section 5.

3 The Proposed Method

In this paper, we would like to evaluate the strategies of transfer learning and fine-tuning on the performance of vehicle classification for a relatively small dataset. Transfer learning can be considered as a shortcut that uses a pre-trained DCNN model to learn patterns from data it was not seen before. The pre-trained model is often trained on a very large-scale dataset such as ImageNet. Therefore, a good pre-trained CNN can be served as a feature extractor with which we can feed our image dataset through the network for extracting the activations (or features) at a given layer.

Fine-tuning can be considered as another type of transfer learning, but its performance is usually better than transfer learning, provided there are sufficient training data. The method of fine-tuning often involves a network modification. First, the FC layers of a pre-trained CNN (also called the "head" of the network) such as VGG [4], ResNet [28], or GoogLeNet [27] are cut off from their body. The head is then replaced by a new set of FC layers with randomly initialized weights. From there, all CNN layers below the head are frozen so the weights cannot be updated, implying that the backward pass in backpropagation cannot reach them.

The new set of FC layers is trained using a very small learning rate until they can learn patterns from the previously learned CNN layers. Afterwards, we may unfreeze the rest of the network, especially for the bottleneck layer, and continue training. The details of transfer learning and fine-tuning on vehicle classification are described in sections 3.1 and 3.2, respectively.

3.1 The Method of Transfer Learning

The task of transfer learning in this paper is conducted in the pre-trained VGG16 network [4] that has been already trained on the ImageNet dataset, which consists of 13 CNN layers and 3 FC layers that were developed by the Visual Geometry Group from Oxford University, as shown in Fig. 1(a). VGG is also awarded the second place on the task of image classification and the first place on localization in ILSVRC-2013. By the VGG16 network, 3×3 convolution kernel and 2×2 maximum pooling are adopted through all the convolutional layers. In this work, we removed the FC layers from the original VGG16 network and used this architecture as a feature extractor, shown in Fig. 1(b), from which the output of the last layer with maximum pooling has a shape of 7×7×512, indicating there are 512 filters each of size 7×7. Hence, we can take these 7×7×512=25,088 values as a feature vector that characterizes the discriminative contents of an image. Since the CNN itself is not capable of recognizing new objects, a traditional machine learning classifier such as linear Support Vector Machine (linear SVM), Logistic regression, or Random Forest is required and connected as a head of the network architecture. In this work, the classifiers of linear SVM and Logistic regression are employed for vehicle classification, respectively, which will be described in more detail later.

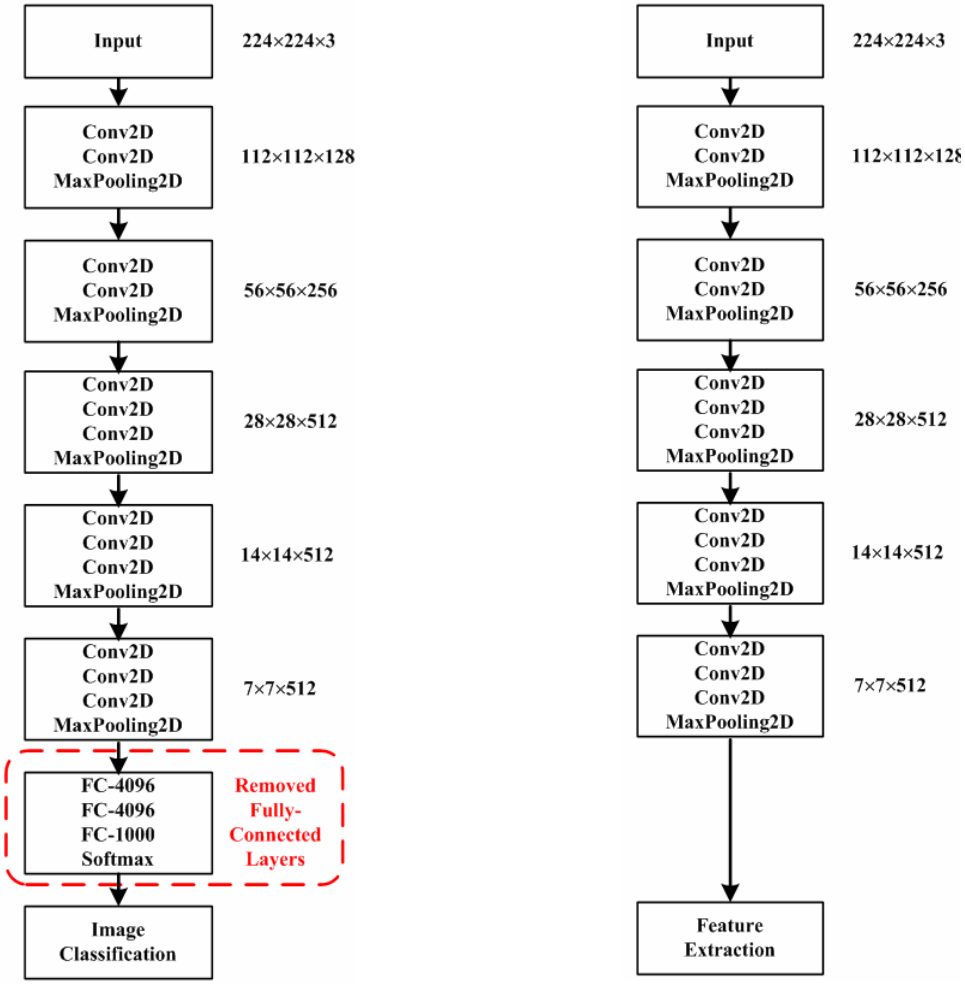
3.1.1 Linear Support Vector Machine

The principle of an SVM classifier is to search for an optimal hyperplane that maximizes the margin of the decision boundary to ensure that the worst-case generalization error is minimized, which is also known as the structural risk minimization.

To perform the classification between two classes, the SVM classifier is applied by transforming the lower-dimensional input space (x_i, y_i) into a higher-dimensional feature space using a kernel $\Phi(x)$, where $x_i \in \Re^d$ (d is the dimension of input vector) and $y_i \in \{+1, -1\}$. Therefore, the optimal hyperplane can be denoted as a decision surface,

$$f(x) = \mathrm{sgn}\left(\sum_i y_i \alpha_i K(x_i, x) + b \right), \tag{1}$$

where sgn(•) represents the sign function, and $K(x_i, x) = \Phi(x_i)^T \Phi(x)$ is a predefined kernel function that satisfies Mercer's condition [39]. In this paper, a linear kernel is used and it is defined as



(a) The original VGG16 network architecture that is used to recognize the 1,000 classes on the large-scale dataset of the ImageNet

(b) Removal of the FC layers from original VGG16 and instead output the features of the final MaxPooling layer. The proposed architecture is served as a feature extractor

Fig. 1.

$$K(x_i, x) = x_i \cdot x, \tag{2}$$

where $x_i \cdot x$ denotes the inner product of feature vectors of x and x_i. The coefficients α_i and b in Eq. (1) can be further determined by the following quadratic programming problem as

$$\max\left[\sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(x_i, x_j)\right], \; s.t. \; \sum_i \alpha_i y_i = 0, \; 0 < \alpha_i < C, \; \forall i. \tag{3}$$

The parameter C is a penalty that represents the tradeoff between minimizing the training set error and maximizing the margin. In this work, C is set to 0.01 empirically. Since the SVM is a binary classifier, it should be extended for a m-class problem in the task of vehicle recognition. We use the so-called one against one approach, which is a pairwise method and needs to train $m(m-1)/2$ SVM classifiers.

3.1.2 Logistic Regression

Logistic regression is a probabilistic statistical classification model for predicting the probability of the occurrence of an event, which is a generalized linear model where the outcome is a two-level categorical variable. Logistic regression creates the relationship between a categorical dependent variable X and a

dichotomous categorical outcome *Y*. The outcome *Y* takes the value 1 (in our case, this represents a given type of vehicle) with probability $p(Y|X)$ and the value 0 with probability $1-p(Y|X)$. The logistic regression can be expressed as

$$p(Y|X)=\frac{1}{1+e^{-z}},\ where\ z=\beta_0+\beta_1 X, \tag{4}$$

$p(Y|X)$ is a logistic function (or sigmoid function). Hence, the generation of the coefficients of the logistic regression can be done by the logit (or log-odds) function, i.e., $\text{logit}(p(Y|X))$, which can be represented as

$$z=\text{logit}(p)=\ln\left(\frac{p(Y|X)}{1-p(Y|X)}\right)=\beta_0+\beta_1 X. \tag{5}$$

Note that the logistic function can receive a range of input values (z or $\beta_0+\beta_1 X$) between negative infinity and positive infinity and the output of $p(Y|X)$ is constrained to values between 0 and 1. The ratio of $p(Y|X)/(1-p(Y|X)$ is also called odds ratio.

The method of logistic regression fits a regression curve using the regression coefficients β_0 and β_1, where the response is a binary variable. As shown in Eq. (4), the logistic function is nonlinear, but the logit function performs a linear regression. Therefore, using the generalization linear model, an estimated logistic regression can be expressed as

$$\text{logit}(p(Y|X_1,...,X_n))=\beta_0+\sum_{k=1}^{n}\beta_k X_k. \tag{6}$$

In this work, the optimization using steepest gradient descent (SGD) with a learning rate of 0.1 is repeatedly iterated by the minimization of the cost function defined as below

$$\text{cost}=\frac{-1}{n}\sum_{k=1}^{n}\Big(y_t\log(y_p)+(1-y_t)\log(1-y_p)\Big), \tag{7}$$

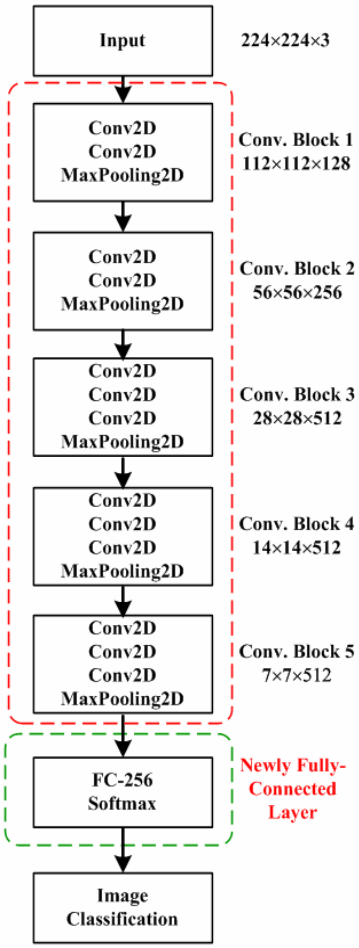
where n is the number of the training samples, y_t and y_p represent the ground truth and predicted value of the sample, respectively.

Logistic regression like the linear SVM is also a binary classifier; therefore, it should be extended for a *m*-class problem in the task of vehicle classification. The training strategy is similar to that used in the linear SVM described earlier, i.e., one-against-one policy.

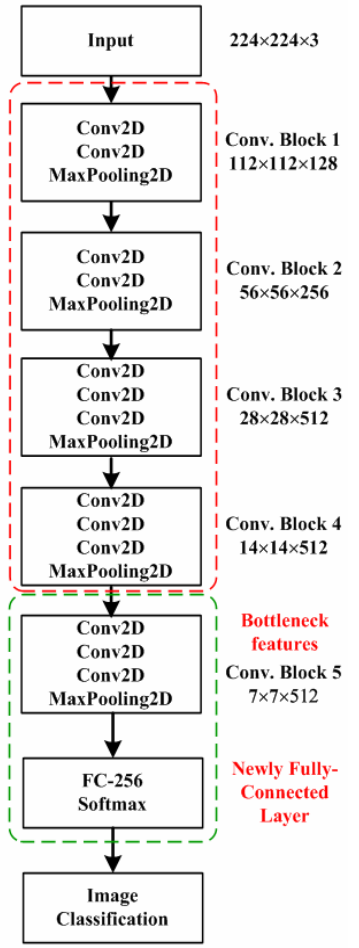
3.2 The Method of Fine-Tuning

In the fine-tuning process, we first replace the FC layers in original VGG16 architecture by a new FC layer with 256 nodes on the top of the network, shown in Fig. 2(a). In this model, the vehicle labels are classified by the softmax layer that outputs the probabilities of recognized objects. We then train the weights of the new FC layer, but freeze the weights in all the CONV layers until the new FC layer can learn the patterns from the pre-trained CONV layers. Why do we have to do this? This is because large gradient updates triggered by the randomly initialized weights of the new FC layer would destroy the weights in the CONV layers pre-trained on the large-scale ImageNet dataset. In our case, this is why we only train the new FC layer but keep the weights in all the CONV layers unchanged.

Once the weights in the new FC layer have been trained, we then unfreeze the weights only in the last CONV block that is just below the top layer, but also keep the weights in the other four CONV blocks unchanged, shown in Fig. 2(b). The features extracted from the last CONV block are also called bottleneck features. Why do we only retrain the weights of the last CONV block? The most important reason is to prevent overfitting. This is because the entire network would have a very large entropic capacity and thus a strong tendency to overfit. As well known, the features learned by the low-level CONV layers are more general, less abstract than those found in the higher ones, so it is suitable to keep the weights in the lower CONV blocks unchanged (more general features) and only fine-tune the last CONV block (more specialized features).



(a) The FC layers in original VGG16 architecture are replaced by a new FC layer with 256 nodes. The green dash box denotes that the weights within it can be trained, but the weights in the red boxes are frozen, indicating that the gradient updates in the backward pass of backpropagation cannot reach them



(b) In the second stage of the fine-tuning process, the weights in the new FC and the last CONV layers can be trained, but the weights in all other four CONV blocks are frozen

Fig. 2.

To have a large dataset is crucial for the performance of the DCNN's models. Data augmentation is often considered as an effective method to improve the performance of the models. In our work, the settings of data augmentation include the horizontal and vertical shift, horizontal flip, random rotation, random zoom, and random shear augmentation. Fine-tuning is usually performed with a very slow learning rate. In the first stage of fine-tuning, the RMSProp optimizer with a learning rate of 0.001 is used. However, in the second stage, the SGD optimizer with a learning rate of 0.001 is utilized in the same case. This is to make certain that the magnitude of the updates could stay very small, so as not to destroy the previously learned features.

4 Results and Discussion

The proposed method of vehicle classification is evaluated on the dataset collected from the Google images. The operation system is Linux Ubuntu 16.04 and the integrated development environment is Python 3.5 64-bit version with installed library packages of Keras 2.1.6, Tensorflow 1.8.0 and Opencv-python 3.4.1.15. The system is running on Intel i7-7770 processor with 16GB memory and one NVIDIA GPU card with a model of GTX 1080Ti having 11GB memory.

The collected images in the training dataset are categorized into four classes of the bus, sedan, sport utility vehicle (SUV) and truck, as shown in Fig. 3. The total amount of collecting images of the bus, sedan, SUV and truck are 684, 973, 712, and 648, respectively. For this dataset, three-quarters of them are used as a training set, and the remaining used as a testing set.

(a) bus

(b) sedan

(c) SUV

(d) truck, respectively

Fig. 3. Typical images in the dataset collected from the Google images, from top to bottom rows are

In the case of transfer learning, the classifiers of the linear SVM and Logistic regression are connected to the top of the VGG16 network architecture, respectively. The metrics of precision, recall, and f1-score used to evaluate the performance of the proposed method are defined in Eqs. (8)-(10), respectively.

$$precision = \frac{TP}{TP + FP}, \tag{8}$$

$$recall = \frac{TP}{TP + FN}, \tag{9}$$

$$f_1 - score = \frac{2}{1/\,precision + 1/\,recall}, \tag{10}$$

where TP and FP denote the outcome being correctly and wrongly identified as positive, respectively, FP and FN represents the outcome being correctly and wrongly identified as negative, respectively. For the metric of precision, it is also called positive predicted value (PPV). But for the metric of recall, it is also called sensitivity, hit rate or true positive rate (TPR). As for the f1-score, it is the average of the reciprocal sum of the metrics of precision and recall.

The classification results and its corresponding confusion matrix are listed in Table 1 and Table 2, respectively. The average recall rate (ARR) is impressive to be about 93%. The erroneously recognized classes are concentrated on the types of SUV and sedan as observed from the distribution of confusion matrix shown in Table 2. This result is not surprising because the sedan and smaller SUVs are similar in appearance, as shown in Fig. 3(b) and Fig. 3(c). The results of classification and confusion matrix conducted by Logistic regression are also listed in Table 3 and Table 4, respectively. As observed from Table 3, the ARR of 93% is the same as that in the case of the linear SVM.

Table 1. Classification results of the linear SVM with C=0.01 in the transfer learning process

	precision	recall	f1-score
SUV	0.91	0.82	0.86
Bus	0.99	1.00	0.99
Sedan	0.86	0.94	0.90
truck	1.00	0.97	0.99
Avg/total	0.93	0.93	0.93

Table 2. Confusion matrix of the linear SVM with C=0.01 in the transfer learning process

	SUV	bus	sedan	truck
SUV	**0.82**	0.00	0.18	0.00
Bus	0.00	**1.00**	0.00	0.00
Sedan	0.06	0.00	**0.94**	0.00
truck	0.01	0.01	0.01	**0.97**

Table 3. Classification results of the Logistic regression in the transfer learning process

	precision	recall	f1-score
SUV	0.91	0.83	0.87
Bus	0.98	0.99	0.99
Sedan	0.87	0.94	0.91
truck	0.99	0.97	0.98
Avg/total	0.93	0.93	0.93

Table 4. Confusion matrix of the Logistic regression in the transfer learning process

	SUV	bus	sedan	truck
SUV	**0.83**	0.01	0.16	0.01
Bus	0.00	**0.99**	0.00	0.01
Sedan	0.06	0.00	**0.94**	0.00
truck	0.01	0.01	0.01	**0.97**

In the case of fine-tuning, we first replaced the FC layers in the original VGG16 network by our new FC layer with 256 nodes, as shown in Fig. 2(a). In order to avoid destroying the weights in the CONV layers pre-trained on the large-scale ImageNet dataset, the weights in all the CONV layers are frozen during the training phase. We call this phase as the first stage in the fine-tuning process. Because DCNN architecture is quite data-hungry, data augmentation is applied during the training phase to avoid overfitting. The settings of data augmentation include rotation angle, width and height shift, shear range, zoom range, and horizontal flip. The training images are randomly generated on the fly based on the settings to avoid memory shortage.

In this work, the cost function of categorical cross-entropy commonly used in the multiclass classification problems is applied and defined as below

$$\text{cost} = \frac{-1}{n}\sum_{k=1}^{n} y_t \log(y_p), \tag{11}$$

where n is the number of the training samples, y_t nd y_p present the label and predicted value of the sample, respectively. In this stage, the RMSProp optimizer with a learning rate of 0.001 is used to iteratively minimize the cost function.

The training curve of accuracy versus epochs is shown in Fig. 4, from which we can see that 25 epochs are sufficient for accuracy to reach a relatively steady value. The definition of accuracy is given in Eq. (12). The results of classification and its corresponding confusion matrix are also given in Table 5 and Table 6, respectively. As revealed by Table 5, the ARR is only 88%, indicating that the weights in the CONV layers need to be further unfrozen.

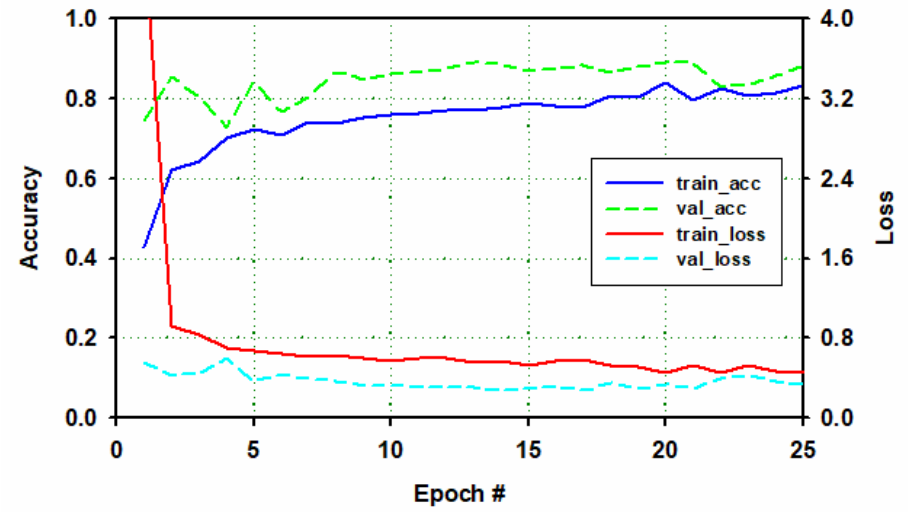


Fig. 4. The training curve of accuracy versus epochs in the first stage of the fine-tuning process

Table 5. Classification results of the first stage in the fine-tuning process

	precision	recall	f1-score
SUV	0.78	0.89	0.83
Bus	0.99	0.90	0.94
Sedan	0.91	0.80	0.85
truck	0.88	1.00	0.94
Avg/total	0.89	0.88	0.88

Table 6. Confusion matrix of the first stage in the fine-tuning process

	SUV	bus	sedan	truck
SUV	**0.89**	0.00	0.10	0.02
Bus	0.01	**0.90**	0.01	0.09
Sedan	0.20	0.00	**0.80**	0.00
truck	0.00	0.00	0.00	**1.00**

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{12}$$

In the second stage of the fine-tuning process, the weights in CONV block 5 shown in Fig. 2(b) are further unfrozen. During this phase, 50 epochs and the same settings of data augmentation as in the first stage are also leveraged. In this stage, the cost function of categorical cross-entropy is also used, which is the same as in the first stage, but instead use the SGD optimizer with a learning rate of 0.001. The training curve of accuracy versus epochs is shown in Fig. 5, from which we can see that 50 epochs are sufficient for accuracy to reach a relatively steady value. The results of classification and its corresponding confusion matrix are also shown in Table 7 and Table 8, respectively. As observed from Table 7, the ARR is increased from 88% in the first stage to 95% in the second stage, indicating that the learning of weights in higher CONV layers is desirable.

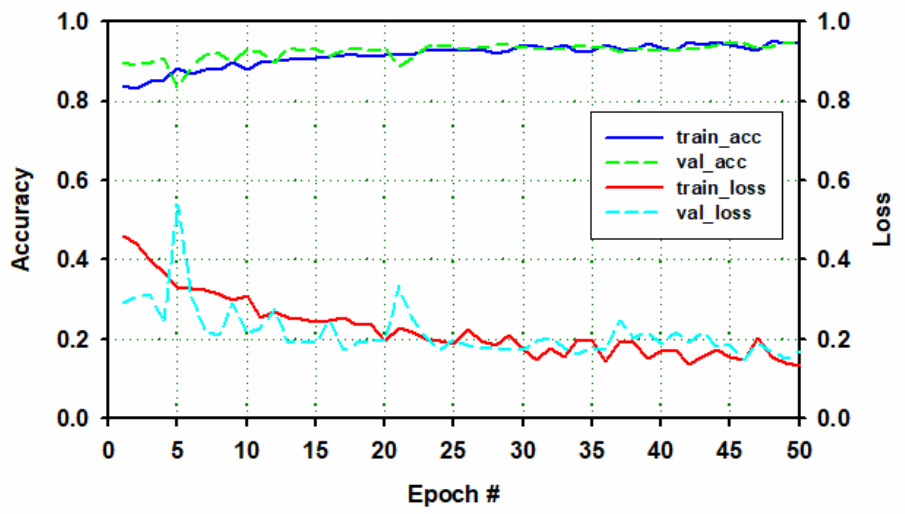


Fig. 5. The training curve of accuracy versus epochs in the second stage of the fine-tuning process

Table 7. Classification results of the second stage in the fine-tuning process

	precision	recall	f1-score
SUV	0.91	0.93	0.92
Bus	0.98	0.98	0.98
Sedan	0.93	0.93	0.93
truck	0.99	0.97	0.98
Avg/total	0.95	0.95	0.95

Table 8. Confusion matrix of the second stage in the fine-tuning process

	SUV	bus	sedan	truck
SUV	**0.93**	0.00	0.07	0.00
Bus	0.00	**0.98**	0.01	0.01
Sedan	0.07	0.00	**0.93**	0.00
truck	0.01	0.02	0.00	**0.97**

5 Conclusion

In this paper, we have made a complete comparison of transfer learning and fine-tuning on the performance of vehicle classification. These two strategies are used for the alleviation of the overfitting problem and the reduction of the cost of data labeling. The images of vehicles are collected from the Google images and categorized into four classes of bus, sedan, SUV, and truck. In the transfer learning process, the linear SVM and Logistic regression are utilized and connected to the top of the VGG16 network, respectively. The average recall rate of 93% is achieved by both the linear SVM and Logistic regression. The impressive result shows that both methods have the same or similar discriminative capability for recognizing different vehicles. In the fine-tuning process, the average recall rate is only 88% in the first stage of freezing the trainable weights in all the CONV layers. But, when we further unfreeze the trainable weights only in the last CONV block, the average recall rate is increased from 88% in the first stage to 95% in the second stage, indicating that the learning of trainable weights in higher CONV layers is highly desirable. As observed from the experiments, fine-tuning with a 95% average recall rate clearly outperform transfer learning with a 93% average recall rate. But the training time on fine-tuning is about one order of magnitude of that for transfer learning. However, for a relatively small dataset, transfer learning is still a good selection due to its short training time and a satisfactory classification rate.

Transfer learning and fine-tuning are very useful in reducing the expensive cost of data labeling and in alleviating the overfitting problem, but however, the limitation of these two strategies is constrained within a tolerant difference between the different domains of interest, i.e., the difference of the feature-

space between different tasks cannot be too large. Therefore, future work will focus on investigating the extent to which the differences between different domains of knowledge can be used in knowledge transfer.

Acknowledgements

This work is completely supported by a grant from the Ministry of Science and Technology (MOST), Taiwan, under contract MOST-107-2221-E-212-012- and MOST-108-2221-E-212-012-.

References

[1] Wikipedia, Deep learning. <https://en.wikipedia.org/wiki/Deep_learning/>, 2020 (accessed 15.01.20).

[2] W. Ballard, Hands-On Deep Learning for Images with Tensorflow: Build Intelligent Computer Vision Applications Using Tensorflow and Keras, first ed., Packt Publishing, Birmingham, UK, 2018 (Chapter 4).

[3] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Advances in Neural Information Processing Systems 25(2012) 1097-1105.

[4] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition. <https://arxiv.org/abs/1409.1556>, 2014 (accessed 10.04.15).

[5] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: integrated recognition, localization and detection using convolutional networks. <https://arxiv.org/abs/1312.6229v4>, 2013 (accessed 24.02.14).

[6] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks. <https://arxiv.org/abs/1506.01497v2>, 2015 (accessed 13.09.15).

[7] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection. <https://arxiv.org/abs/1506.02640v5> 2015 (accessed 09.03.16).

[8] E. Shelhamer, J. Long, T. Darrell, Fully convolutional networks for semantic segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39(4)(2017) 640-651.

[9] V. Badrinarayanan, A. Kendall, R. Cipolla, SegNet: A deep convolutional encoder-decoder architecture for image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 39(12)(2017) 2481-2495.

[10] K. He, G. Gkioxari, P. Dollar, R. Girshick, Mask R-CNN. <https://arxiv.org/abs/1703.06870v3> 2018 (accessed 24.01.18).

[11] L.A. Gatys, A.S. Ecker, M. Bethge, A neural algorithm of artistic style. <https://arxiv.org/abs/1508.06576v2>, 2015 (accessed 02.09.15).

[12] L.A. Gatys, A.S. Ecker, M. Bethge, Image style transfer using convolutional neural network, in: Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[13] R. Zhang, P. Isola, A.A. Efros, Colorful image colorization. <https://arxiv.org/abs/1603.08511v5>, 2016 (accessed 05.10.16).

[14] Z. Cheng, Q. Yang, B. Sheng, Deep colorization. <https://arxiv.org/abs/1605.00075v1>, 2016 (accessed 30.04.16).

[15] A. van den Oord, N. Kalchbrenner, K. Kavukcuoglu, Pixel recurrent neural networks. <https://arxiv.org/abs/1601.06759v3>, 2016 (accessed 19.018.16).

[16] J.Y. Cheng, F. Chen, M.T. Alley, J.M. Pauly, S.S. Vasanawala, Highly scalable image reconstruction using deep neural networks with bandpass filtering. <https://arxiv.org/abs/1805.03300v2>, 2018 (accessed 26.11.18).

[17] W.-S. Lai, J.-B. Huang, N. Ahuja, M.-H. Yang, Deep Laplacian pyramid networks for fast and accurate super-resolution. <https://arxiv.org/abs/1704.03915v2>, 2017 (accessed 09.10.17).

[18]D. Ulyanov, A. Vedaldi, V. Lempitsky, Deep image prior. <https://arxiv.org/abs/1711.10925v3>, 2018 (accessed 05.04.18).

[19] A. Radford, L. Meta, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks. <https://arxiv.org/abs/1511.06434v2>, 2016 (accessed 07.01.16).

[20] A. van den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves, K. Kavukcuoglu, Conditional image generation with pixelCNN decoders <https://arxiv.org/abs/1606.05328v2>, 2016 (accessed 18.01.16).

[21] R. Sarikaya, G.E. Hinton, A. Deoras, Application of deep belief networks for natural language understanding, IEEE/ACM Transactions on Audio, Speech, and Language Processing 22(4)(2014) 778-784.

[22] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 81(11)(1998) 2278-2324.

[23] Y. Qian, M. Bi, T. Tan, K. Yu, Very deep convolutional neural networks for noise robust speech recognition, IEEE Transactions on Audio, Speech, and Language Processing 24(12)(2016) 2263-2276.

[24] H. Fukui, T. Yamashita, Y. Yamauchi, H. Fujiyoshi, H. Murase, Pedestrian detection based on deep convolutional neural network with ensemble inference network, in: Proc. 2015 IEEE Intelligent Vehicle Symposium (IV), 2015.

[25] H. El Abed, V. Margner, M. Kherallah, A. M. Alimi, ICDAR 2009 online Arabic handwriting recognition competition, in: Proc. 2009 IEEE International Conference on Document Analysis and Recognition, 2009.

[26] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks. <https://arxiv.org/abs/1311.2901v3>, 2013 (accessed 28.11.13).

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions. <https://arxiv.org/abs/1409.4842v1>, 2014 (accessed 17.09.14).

[28] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition. <https://arxiv.org/abs/1512.03385v1>, 2015 (accessed 10.12.15).

[29] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on Knowledge and Data Engineering 22(10)(2010) 1345-1359.

[30] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural network? in: Proc. Advances in Neural Information Processing Systems (NIPS), 2014.

[31] S.Y. Jo, N. Ahn, Y. Lee, S.J. Kang, Transfer learning-based vehicle classification, in: Proc. 2018 International SoC Design Conference, 2018.

[32] C. Li, Y. Bai, Fire flame image detection based on transfer learning, in: Proc. 2018 5th IEEE International Conference on Cloud Computing and Intelligence Systems, 2018.

[33] C. Qiu, J. Cui, S. Zhang, C. Wang, Z. Gu, H. Zheng, B. Zheng, Transfer learning for small-scale fish image classification, in: Proc. 2018 OCEANS-MTS/IEEE Kobe Techno-Oceans (OTO), 2018.

[34] Y. Wu, X. Qin, Y. Pan, C. Yuan, Convolution neural network based transfer learning for classification of flowers, in: Proc. 2018 IEEE 3rd International Conference on Signal and Image Processing, 2018.

[35] P. Sreekanth, U. Kulkarni, S. Shetty, S.M. Meena, Head pose estimation using transfer learning, in: Proc. 2018 International Conference on Recent Trends in Advance Computing, 2018.

[36] B. Ghosh, M.K. Bhuyan, P. Sasmal, Y. Iwahori, P. Gadde, Defect classification of printed circuit boards based on transfer learning, in: Proc. 2018 IEEE Applied Signal Processing Conference, 2018.

[37] W. Liu, R. Lu, X. Liu, Traffic sign detection and recognition via transfer learning, in: Proc. 2018 Chinese Control and Decision Conference, 2018.

[38] A.S. Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognition. <https://arxiv.org/abs/1403.6382v3>, 2014 (accessed 12.05.14).

[39] V. N. Vapnik, Statistical learning theory, John Wiely and Sons, New York, 1998, pp. 423-424.