LG-CNN: From local parts to global discrimination for fine-grained recognition

Guo-Sen Xie\textsuperscript{a,}\textsuperscript{,}, Xu-Yao Zhang\textsuperscript{a}, Wenhuan Yang\textsuperscript{c}, Mingliang Xu\textsuperscript{d}, Shuicheng Yan\textsuperscript{e}, Cheng-Lin Liu\textsuperscript{a,}\textsuperscript{b,}\textsuperscript{*}

\textsuperscript{a} National Laboratory of Pattern Recognition, Institute of Automation of Chinese Academy of Sciences, Beijing 100190, China
\textsuperscript{b} University of Chinese Academy of Sciences, Beijing, China
\textsuperscript{c} Peking University, Beijing 100871, China
\textsuperscript{d} Center for Interdisciplinary Information Science Research, Zhengzhou University, Zhengzhou 450001, China
\textsuperscript{e} National University of Singapore, Singapore 119077, Singapore

\section*{A R T I C L E   I N F O}

\begin{footnotesize}
\begin{tabular}{ll}
Article history: & \\
Received 12 July 2016 & \\
Revised 17 May 2017 & \\
Accepted 1 June 2017 & \\
Available online 3 June 2017 & \\
\end{tabular}
\end{footnotesize}

\begin{footnotesize}
\begin{tabular}{ll}
Keywords: & \\
Fine-grained recognition & \\
Convolutional neural networks & \\
Bilinear pooling & \\
Local parts & \\
Global discrimination & \\
\end{tabular}
\end{footnotesize}

\section*{A B S T R A C T}

Fine-grained recognition is one of the most difficult topics in visual recognition, which aims at distinguishing confusing categories such as bird species within a genus. The information of part and bounding boxes in fine-grained images is very important for improving the performance. However, in real applications, the part and/or bounding box annotations may not exist. This makes fine-grained recognition a challenging problem. In this paper, we propose a jointly trained Convolutional Neural Network (CNN) architecture to solve the fine-grained recognition problem without using part and bounding box information. In this framework, we first detect part candidates by calculating the gradients of feature maps of a trained CNN w.r.t. the input image and then filter out unnecessary ones by fusing two saliency detection methods. Meanwhile, two groups of global object locations are obtained based on the saliency detection methods and a segmentation method. With the filtered part candidates and approximate object locations as inputs, we construct the CNN architecture with local parts and global discrimination (LG-CNN) which consists of two CNN networks with shared weights. The upper stream of LG-CNN is focused on the part information of the input image, the bottom stream of LG-CNN is focused on the global input image. LG-CNN is jointly trained by two stream loss functions to guide the updating of the shared weights. Experiments on three popular fine-grained datasets well validate the effectiveness of our proposed LG-CNN architecture. Applying our LG-CNN architecture to generic object recognition datasets also yields superior performance over the directly fine-tuned CNN architecture with a large margin.

© 2017 Elsevier Ltd. All rights reserved.

\section*{1. Introduction}

Recent years have witnessed great progress in visual recognition. For example, the accuracy on the large scale visual recognition challenge (ILSVRC12) has surpassed human-level performance [1]. Researchers are turning to solve more complex and difficult visual recognition tasks, e.g., large scale scene classification, fine-grained recognition [2,3]. A traditional system for image categorization is the Bag of Word (BoW) model [4,5], which consists of local descriptor extraction [6,7], feature coding [8–11], feature pooling [12,13], and classifier training. The Fisher vector (FV) based recognition system is another successful model [14–16], which can also be viewed as a BoW model by viewing the FV as a coding method. To speed up the FV method, Fisher codemap [17], which can work with arbitrary lattices for images, is further proposed. From the year of 2012, Convolutional Neural Networks (CNNs) [18] have becoming the leading system for visual recognition [19–23]. Remarkable performances have been achieved for many famous medium-scale recognition tasks [24–26] based on the CNN models trained on ImageNet database. Moreover, CNNs also achieve state-of-the-art performances in many other tasks such as object detection [27–31], image segmentation [32–34] and image retrieval [35].

Fine-grained recognition is one of the most difficult tasks in the community of visual recognition. Specifically, the fine-grained categories are difficult to discriminate due to the small inter-class difference and the large intra-class variation [36,37]. With the successful applications of CNNs to the large-scale recognition challenge [18], fine-grained recognition has drawn increasing atten-
tion in recent years, and the performance has been improved remarkably, e.g., from about 20% [38] to 84.0% [2] on the CUB-200 dataset [36]. As far as we know, the CNN recognition system is the key to achieve the breakthroughs in all these fields. According to the availability of part information, there exist three settings for the fine-grained recognition problem, i.e., (I) utilizing the bounding box and/or part information during both training and testing phase, (II) utilizing the bounding box and/or part information during training phase, but not during testing phase, and (III) utilizing no bounding box and part information during both training and testing phase, which is also called weakly supervised fine-grained recognition and is the most difficult setting. In the real world, annotations of bounding box and part information is very time-consuming and economically expensive, either in training or testing. In this paper, we will focus on this difficult setting (utilizing no bounding box and part information in both training and testing phase) for fine-grained recognition.

Our proposed fine-grained recognition system consists of two stages: (1) unsupervised part candidate and object bounding box discovery, and (2) joint training of CNN with Local parts and Global discrimination (LG-CNN training). As for the part candidate discovery, we use the method of calculating the gradients of the last convolutional feature maps w.r.t. the input image [39]. The segmented images are taken as the global inputs of our LG-CNN architecture. The LG-CNN network architecture is illustrated in Fig. 1, which shows that (1) the network takes the original image with part candidate boxes as the input of the upper sub-network; (2) the salient images and the segmented images are taken as inputs to the bottom sub-network; (3) the weights of the two sub-networks are shared until the last convolutional layer; (4) the locally max-pooled representations on the last convolutional layer of the upper sub-net are straightened as one long vector. After training the networks, we further construct a Fisher vector representation of the last convolutional layer. The final prediction is made by fusing these several parts. Experiments on three popular fine-grained datasets well validate our proposed method. When transferring the LG-CNN framework to generic object recognition, improvement over baseline fine-tuned CNN is still evident.

The reminder of this paper is organized as follows. Section 2 introduces related works, Section 3 describes the proposed fine-grained recognition system. Section 4 presents our experimental results, and Section 5 offers concluding remarks.

2. Related works

As discussed in Section 1, fine-grained recognition methods can be categorized into three settings. Before CNNs become the leading recognition system, many approaches are proposed to solve the fine-grained recognition problem under the settings of (I) and (II). For example, Zhang et al. [41] advocate to use deformable part-based model for detecting objects so as to solve the fine-grained recognition problem. Rubio et al. [42] propose one generative regularization with latent topics, while training the part-based model. Berg and Belhumeur [43] present POOF (Part-based One-vs-One Feature), which is constructed by specializing in discrimination between two particular categories, based on the appearance at a particular part. Göring et al. [44] discover a nonparametric label transfer technique to transfer parts from objects with similar global shapes. Similar to that work, Gavves et al. [45] also try to transfer part locations from training images to testing images by nearest HOG shape searching. Chai et al. [46] advocate to model the segmentation and part localization in a joint manner. The above methods all belong to the category (I) for fine-grained recognition, and they are traditional methods which are not based on the CNN model. Due to the success of CNN on object recognition (year 2012), many fine-grained recognition methods based on CNN are explored. The most famous method is the part-based R-CNN (Part RCNN) proposed by Zhang et al. [47], the fine-grained recognition accuracy has been improved to a new level based on Part RCNN. Then Branson et al. [48] propose pose normalized CNN to compute warped image region features, and achieve better results than Part RCNN. Lin et al. [49] propose Deep LAC, which incorporates part localization, alignment, and classification into one CNN model. Recently, Krause et al. [50] try to solve fine-grained recognition without part annotations, by generating parts through co-segmentation and alignment. Very recently, unlike previous step-by-step method, Zhang et al. [51] utilize fully convolutional network to detect parts, and then train an end-to-end network supervised by these parts.
and labels. However, this method pools their features on the final fully connected layer (transferred as convolutional layer), resulting in very high dimensional middle representations (4096 × P. P is the number of parts). Another work related to ours is the part-stacked CNN [52], which also utilize fully convolutional network to localize object parts, then the boxes centering at the part points are taken out and stacked into multiple small feature maps together with the original feature maps, followed by one stream classifier.

As for category (III) of fine-grained recognition, an initial work is the two-level attention model [53], which conducts the part-level detection and the object-level filtering, and the final prediction is the fusion of the part-level and object-level classifiers. Another work is the multiple granularity descriptor (MGD) [54] which is based on the ontology tree to train three CNNs, specialized at three grain levels. Simon and Rodner [55] present to find the neural activation (from CNN) constellations for fine-grained recognition. To allow the spatial invariant manipulation of input data, Spatial Transformer Network (STN) [3] is proposed for fine-grained recognition, which is based on inception architecture of CNN (a localization network followed by two inception nets). STN first predicts the transformation parameters of the fine-grained images, followed by the common operations in CNN. Recently, bilinear CNN (BCNN) [2] is proposed to solve the fine-grained problem, BCNN is a two stream CNN architecture, wherein the bilinear vector is calculated from the last convolutional feature maps of the input two stream CNNs. However the dimensionality of the bilinear feature from BCNN is usually very high, e.g., if the number of feature maps is 512, the dimensionality of bilinear feature will be about 260,000. BCNN has achieved excellent performance on the fine-grained recognition problem. To solve the high dimensionality problem, the compact bilinear pooling method [56] is proposed, which can reduce the dimensionality while preserving the accuracy at the same time.

Our proposed LG-CNN architecture is different from previous works in the following four aspects: (1) we do not utilize bounding box and part annotations during training and testing phase; (2) two stream network architecture is constructed with weight sharing until the last convolutional layer; (3) parts at the top stream are max pooled and stacked (ordered by the magnitude of gradients w.r.t. the part points) at the last convolutional layer; (4) data augmentation of the bottom stream is implemented by saliency and segmentation methods. Overall, we seek one strategy to solve the problem of category (III) for fine-grained recognition.

3. The proposed fine-grained recognition system

In this section, we first show our LG-CNN network architecture for fine-grained recognition, then describe the techniques of data collection for training networks and feature representations construction for classification.

3.1. LG-CNN architecture

As shown in Fig. 1, left part of the dotted purple line is our data preprocessing phase, right part of the dotted purple line is the architecture of LG-CNN. In this section, we suppose the data has been processed (in Section 3.2), we will show the details of data preprocessing), and mainly show the network architecture of our system. In the right part of Fig. 1, the upper stream is used for catching the local parts of the fine-grained images, while the bottom stream is used for keeping the original global information. Specifically, the inputs to the upper stream are the resized original images (all images are resized to 256 × 256, without using the mirrors of them) and the coordinates of local boxes (discovered by the unsupervised method described in Section 3.2), then features in the boxes of the last convolutional layer corresponding to the input local boxes are max-pooled and straightened into a long vector. Finally, the straightened vector is connected to a soft-max loss function, for the purpose of predicting the labels of the input images. For example, in Fig. 1, the three green boxes in the last feature maps correspond to the input three green boxes in the original image, and the max-pooled and straightened vector is one 512 × 3 dimensional vector, then followed by the C-way (C is the number of categories) soft-max layer loss.

As for the bottom stream of LG-CNN, the input images are the same as original VGG-16 network [19], with image size of 224 × 224, which are randomly cropped from original images and their mirrors. After the last convolutional layer, two 4096 dimensional fully connected layers are followed, finally C-way soft-max layer are used for supervision. Therefore, the bottom stream is the same as the initial VGG-16 network [19] architecture except the final number of output, which is used for the transferred classification task. Note that in both the upper stream and bottom stream, we add one normalization layer to make the activated values of the two streams consistent with each other. Specifically, we carry out tanh activating operation on the feature maps before the straighten layer in the upper stream and conduct tanh activating on the fully connected layer before the last fully connected layer in the bottom stream, respectively.

The weights before the last convolutional layer (marked as blue in Fig. 1) of LG-CNN are shared. By weight sharing, we can save the storage of model parameters. On the other hand, the discrimination from local parts can be gradually transferred to global features. Given the processed training data, the forward pass of our network is easy to carry out, while for the backward pass, the gradient updating depends on two parts, i.e., the upper stream loss and the bottom stream loss. Let the data samples of the upper stream be \{ (x_i, y_i) \}_{i=1}^M, y_i \in \{1, 2, \cdots, K\} (K is the number of categories), and M is the total number of the data. Meanwhile, let the data samples of the bottom stream be \{ (z_i, g_i) \}_{i=1}^N,\ where \ g_i \in \{1, 2, \cdots, K\} (K is the number of categories), and N is the number of the data. Then the final loss function of the two parts is as follows:

$$L = L_u + \lambda L_b = \frac{-1 \sum M \sum K \mathbf{1}(y_i = j) \log p_{ij}}{\lambda N \sum N \sum K \mathbf{1}(g_i = j) \log q_{ij}}$$

(1)

where \(\mathbf{1}(\cdot)\) is the indicator function, \(\lambda\) is used to control the trade-off between the two losses, and

$$p_{ij} = \frac{\exp(u_{ij})}{\sum_{l=1}^{K} \exp(u_{il})}$$

(2)

$$q_{ij} = \frac{\exp(b_{ij})}{\sum_{l=1}^{K} \exp(b_{il})}$$

(3)

Here, \(u_i = [u_{i1}, u_{i2}, \cdots, u_{iK}]^T \in \mathbb{R}^{K \times 1}\) in Eq. (2) and \(b_i = [b_{i1}, b_{i2}, \cdots, b_{iK}]^T \in \mathbb{R}^{K \times 1}\) in Eq. (3) are the outputs of the last layer of the upper and bottom stream of LG-CNN w.r.t. input image \(x_i\) and \(z_i\), respectively. In practice, the gradient updating of the network parameters is based on batch data, i.e., the number (M and N) of samples in Eq. (1) is usually taken as the number of the batch data.

To calculate the gradients of L w.r.t. the parameters of LG-CNN, we first need to deduce the gradients of L w.r.t. \(u_{ij}\) and \(b_{ij}\) as follows:

$$\frac{\partial L}{\partial u_{ij}} = \frac{-1}{M} (1 - \exp(u_{ij}))$$

(4)
\[
\frac{\partial L}{\partial b_{ij}} = -\lambda \frac{1}{N} (1 - \exp(q_{ij})).
\]

Then the gradient updating after the last shared convolutional layer of LG-CNN for the upper stream and bottom stream comes from the back propagation of the gradients from Eqs. (4) and (5), respectively. As for the gradient updating of the shared weights (before the last convolutional layer) of LG-CNN, both the gradients from Eqs. (4) and (5) are utilized.

### 3.2. Training data collection

In this subsection, we will show our unsupervised training data collection stage, we first show the proposed unsupervised part localization and filtering method, followed by the unsupervised global object discovery method.

#### 3.2.1. Unsupervised part localization

In this part, we will describe the detailed unsupervised part localization procedure. As shown in Fig. 2, given the input image \(I\), the procedure of the part box localization is as follows:

(1) **Part candidates generation:** We utilize the method proposed by Simonyan [39,55] to get the redundancy part candidates (red points in the middle image of Fig. 2(b)). Specifically, based on the AlexNet [18] (trained on ImageNet dataset), we calculate the gradients of each of the last convolutional feature maps w.r.t. the input image \(I\). There are totally 256 feature maps in the last convolutional layer of AlexNet, thus generating 256 part candidates. Suppose the feature maps are \(F_i\), \(i = 1, 2, \ldots, 256\), then the gradient map \(G_i \in \mathbb{R}^{256 \times 256 \times 3}\) of \(F_i\) w.r.t. the input image \(I\) can be obtained through back propagation of \(F_i\) [39,55]. Furthermore, we calculate the sum of absolute values for the three channels of \(G_i\) as follows:

\[
\bar{G}_i = \frac{1}{3} \sum_{l=1}^{3} |G_i^{(l)}|,
\]

where \(G_i^{(l)} \in \mathbb{R}^{256 \times 256}, (l = 1, 2, 3)\) is the gradient of channel \(l\) from \(G_i\). Then we can get the ith part candidate corresponding to image \(I\) by finding the position with maximum value in \(\bar{G}_i\). We also record the maximum value \(v_i\) in \(\bar{G}_i\). The final coordinates and its corresponding maximum value \(v_i\) w.r.t. \(G_i\) can be written as a triple \(V_i = (y_i, x_i, v_i)\). Similarly, we can get all the 256 triples \(V = [v_1; v_2; \ldots; v_{256}] \in \mathbb{R}^{256 \times 3}\) w.r.t. the 256 feature maps.

(2) **Saliency map generation:** Given the input image \(I\), we further generate its two saliency maps (upper and bottom images of Fig. 2(b)). Here we use two saliency detection methods, i.e., the discriminative regional feature integration approach (DRFI) [57] and the hierarchical saliency (HS) detection method [58]. After generating the two saliency maps, we fuse them together by summing and dilating (bottom image of Fig. 2(c)).

(3) **Part candidates filtering:** As can be seen in Fig. 2, there exist many redundancy part candidates in \(V\). Based on the fused saliency map, we generate its mask, which is used for filtering out unnecessary part candidates. Specifically, we sort the part coordinates (first two columns in \(V\)) by descending order according to the values of the third column in \(V\). The part candidates are preserved only if its coordinates are within the mask and we take the top \(P\) (taken as 20 in this paper) candidates according to their values (in descending order) in the third column of \(V\). The red points in Fig. 2(d) are the preserved part candidates.

(4) **Part box generation:** After obtaining the filtered part candidates, we further calculate the top-left and bottom-right coordinates of the generated boxes, which are generated by taking the coordinates of part candidates as the centers (See Fig. 2(e) for il-
lustration). During the training phase of LG-CNN, the coordinates of the boxes are also fed into the network to record the part locations, used for the later local max-pooling in the last convolutional layer of the upper stream. The straighten layer orders the max-pooled vectors according to the descending order of their values in the 3rd column of V.

3.2.2. Unsupervised object discovery

In this part, we describe the data augmentation procedure of the bottom stream in LG-CNN. We first utilize the saliency detection methods in Section 3.2.1 to generate one group of cropped global images. Then, by viewing the fused saliency map as the seed of GSC segmentation method [40], we carry out segmentation on the resized input image, and generate another group of cropped global images.

1) Object discovery by saliency detection: As can be seen in Fig. 3, after obtaining the two saliency maps and their corresponding cropped saliency images (resized into squared images, in Fig. 3(c)), we further select the larger (with more pixels) saliency image. In Fig. 3, the cropped image by HS is selected. Finally, the global object w.r.t. the bounding box of the selected image is extracted, enlarged by several pixels and resized into squared box, which will be taken as inputs for the bottom stream in LG-CNN.

2) Object discovery by segmentation: In the process of generating global object by saliency detection methods, some generated images fail to capture the whole objects (See Fig. 5 for comparisons, it can be seen that the global objects discovered by saliency detection are usually incomplete, while the global objects by segmentation are complete) due to the unliability of the saliency detection approaches. We propose to carry out GSC segmentation by viewing the additive saliency map as the initial seeds (Fig. 4(c)) for segmentation. Specifically, GSC [40] introduced geodesic (multiple) star convexity (a new shape constraint) for image segmentation task. Here we imitate the human interaction of providing brush strokes by generating the unknown, foreground (FG) and background (BG) random seeds through processing the additive saliency map. Pixels with higher values are regarded as foreground seeds, whereas lower value areas are background seeds, and the rest areas are labeled as unknown. Finally, we generate foreground segmentation by GSC, which can produce more accurate locations of the included objects. Then the segmented mask is obtained. Based on the bounding box of the mask, global object is cropped and resized into squared box (Fig. 4(e)), which will be also taken as inputs of the bottom stream in LG-CNN.

3.3. Feature representation

In this subsection, we will illustrate the feature representations used for the final classification, after training the LG-CNN. We use four types of features extracted from the trained LG-CNN. The first feature is the global feature (extracted from the last fully connected layer of the bottom stream in LG-CNN, denoted as “global”). The second feature is obtained from the straighten layer (upper stream in LG-CNN, denoted as “local”), which is used to capture the local information of the input image. The third features are obtained by taking the enlarged part boxes (discovered in Section 3.2.1) as the inputs of the bottom stream of LG-CNN, then the “global” representations of these boxes are max-pooled (ave-pooled) into one vector (denoted as “fc_max” (fc_ave)) (see Fig. 6 for illustrations). The fourth feature is the Fisher vector representation (denoted as “fv”) [59] of the last convolutional layer of LG-CNN. After we get the above feature representations, the linear SVM is used to train classifier and conduct prediction. In practice, we found that late fusion of fv with other feature representations can achieve better performance, thus we report the performances by late fusion for the cases of fv combined with other feature representations.

Fisher vector: In this part, we further review the Fisher vector representation. Suppose we are given the multi-scale activates \( \{X^{(i)}|X^{(i)} \in \mathbb{R}^{d \times N^{(i)}}\} \), \( X^{(i)} = [x^{(i)}_1, x^{(i)}_2, \ldots, x^{(i)}_{N^{(i)}}] \), \( i = 1, 2, \ldots, 5 \) from all the training images. Let \( \rho_i = \sum_{t=1}^{M} \omega_t \rho_i(x) \) denote a Gaussian Mixture Model (GMM), where \( \eta = \{\eta_i = (\omega_i, \mu_i, \sigma_i), \mu_i = 1, 2, \ldots, M\} \) represents the parameters of the GMM, and \( \eta \) can be optimized by the Maximum Likelihood (ML) estimation based on \( \{X^{(i)}, s = 1, 2, \ldots, S\} \). Denote the multi-scale activates of the image \( x \) as \( \chi = \{\chi_i = [x^{(i)}_1, x^{(i)}_2, \ldots, x^{(i)}_{N^{(i)}}], s = 1, 2, \ldots, S\} \). The gradients of
Fig. 4. The unsupervised object discovery procedure based on segmentation: (a) the given input image, (b) saliency maps, (c) the seeds for segmentation, (d) the segmentation mask, and (e) the cropped global object (resized to squared image). Best viewed in color.

Fig. 5. Comparisons of (a) images generated by segmentation, (b) images generated by saliency detection. Images are from CUB-200 datasets. Best viewed in color.

Fig. 6. An example of extracting fc_ave (fc_max) features. Given the input image with three part boxes, the part boxes are resized into $224 \times 224$ and taken as inputs to LG-CNN, then the global features of them are extracted. ave or max pooling is further carried out on these global features. The final output is the fc_ave or fc_max features. Best viewed in color.

The Fisher vector of the image $I$ is obtained by concatenating all the gradients w.r.t. those $M$ Gaussians. In [60], while calculating gradients w.r.t. each Gaussian, Yoo et al. adopted Multi-scale Pyramid Pooling, where first the GMM parameters are generated based on all the descriptors from different scaled training images, and then scale-specific normalization and max-pooling are implemented. In this paper, the fv representation is also constructed under the multi-scale settings.

**Compared baseline features:** recently, the frequently used baseline feature from CNN model is the fully connected layer representation (abbreviated to “global baseline”), which is obtained by
forward propagating the input image until the last fully connected layer of the fine-tuned CNN. Another baseline feature is the Fisher vector representation (abbreviated to “fv baseline”) from the fine-tuned CNN model. Here, the fine-tuned CNN model is obtained by directly fine-tuning the original VGG-16 net (trained from ImageNet dataset).

4. Experiments

To evaluate the performance of the proposed method, we conducted experiments on three popular fine-grained datasets, i.e., CUB-200 [36], Flower-102 [61] and Pets-37 [62]. We performed data preprocessing using Matlab, and implemented LG-CNN training using the Caffe toolbox [63] on a GPU server with 2 Intel i7-5960X CPUs 3.0GHz with 16 cores. In all these three datasets, we report the mean class-wise accuracy.

4.1. Parameter settings

In all our experiments, the basic CNN network is VGG-16 network [19]. The initial shared convolutional layer parameters of LG-CNN are the same as the parameters of VGG-16 (trained from ImageNet dataset). The parameters for the first two fully connected layers (4096 dimensionality) in the bottom stream of LG-CNN are taken from VGG-16, and the last fully connected layer is randomly initialized from Gaussian distribution. The straighten layer and last fully connected layer of the upper stream of LG-CNN are initialized from Gaussian distribution as well. The trade off parameter in Eq. (1) is set as 0.5 in all these three datasets. The preserved part box number in Section 3.2.1 is set as 20, considering that objects in images of CUB-200 and Flower-102 usually occupy a small region of the whole image, we add another five boxes into the upper stream during the training of LG-CNN for CUB-200 and Flower-102. The added 5 boxes are one cropped global object box by saliency detection, and four 2 x 2 spatial partitioned boxes on original image. The size of the 20 part boxes are 50 × 50 in all these three datasets. As for the initialization of LG-CNN, we utilize the same setting as the initial VGG-16 net, except the final layers. For CUB-200, its learning rates (lr) for the two streams are both set as 0.0001 for all the layers. For Flower-102, the upper stream lr is 0.0001 for all the layers, and the bottom stream lr is set as 0.001 for the last layer, 0.0001 for other layers. For Pets-37, the lr is 0.001 for the last layer and 0.0001 for other layers, for both the upper and bottom stream of LG-CNN. In all these three datasets, lr is reduced to one tenth of the current rates after fixed iterations (10,000 in our experiments). The total iterations for three datasets are 50,000. For fc_max and fc_ave representations, four sizes of enlarged input boxes are utilized, i.e., 128 × 128, 150 × 150, 180 × 180 and 200 × 200, resulting in the following features: fc_max1 (fc_ave1), fc_max2 (fc_ave2), fc_max3 (fc_ave3) and fc_max4 (fc_ave4).

As for the parameters of fv [64], we use the same strategy as [59]. The Gaussian components of GMM training is fixed as 64, and multiple scales are used in this paper. Specifically, given the CNN input size of L × L for the image l, the used five scales are L × [2^1234] = [L × 2^1 2^1L 2^2L 2^3L 4L].

The parameter C of linear SVM is set as 1 in all these experiments.

4.2. CUB-200 experiments

The CUB-200 dataset [36] is a famous fine-grained dataset, many algorithms have been proposed and evaluated on it. CUB-200 includes 11,788 images of 200 bird species. Images from CUB-200 are labeled with their categories, object bounding boxes and fifteen part candidate points. In our experiments, we only use the image labels. We use the available training-test partition [36], i.e., 5994 images for training and 5794 images for testing. We first give the performances of each feature representations and several of their combinations. Note that we also use mirror representation w.r.t. fc (denoted as fc_mirror) for CUB-200.

From Table 1, it can be seen that our final result by fusion is 81.8%, which is competitive compared with other state-of-the-art methods. The 4096 dimensional global feature has achieved 78.0% accuracy, which is much better than the baseline (70.4% [2]) of directly fine-tuning VGG-16. Furthermore, the fv representation based on LG-CNN is also much better than the counterpart fv representation baseline (74.7% [2]) constructed based on fine-tuned VGG-16.

In Table 2, we list the results of other state-of-the-art methods. It can be seen that after the fusion of our representation with the bilinear representation [2], we achieve 86.1% accuracy, which is the

Table 1
Classification rates under various types of features on CUB-200, LG-CNN of VGG-16.

<table>
<thead>
<tr>
<th>Features</th>
<th>Acc (%)</th>
<th>Features</th>
<th>Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>global baseline</td>
<td>70.4</td>
<td>fv baseline</td>
<td>74.7</td>
</tr>
<tr>
<td>local</td>
<td>73.5</td>
<td>global+local</td>
<td>79.2</td>
</tr>
<tr>
<td>fc_max1</td>
<td>77.2</td>
<td>global+local+fc_max1</td>
<td>79.6</td>
</tr>
<tr>
<td>fc_ave1</td>
<td>76.9</td>
<td>global+local+fc_ave1</td>
<td>79.8</td>
</tr>
<tr>
<td>fc_max2</td>
<td>77.8</td>
<td>global+local+fc_max2</td>
<td>79.9</td>
</tr>
<tr>
<td>fc_ave2</td>
<td>77.8</td>
<td>global+local+fc_ave2</td>
<td>79.4</td>
</tr>
<tr>
<td>fc_max3</td>
<td>78.0</td>
<td>global+local+fc_max3</td>
<td>79.3</td>
</tr>
<tr>
<td>fc_ave3</td>
<td>77.5</td>
<td>global+local+fc_ave3</td>
<td>79.6</td>
</tr>
<tr>
<td>fc_max4</td>
<td>77.2</td>
<td>global+local+fc_ave4</td>
<td>79.2</td>
</tr>
<tr>
<td>fc_mirror</td>
<td>77.6</td>
<td>global+local+fc_max1+fc_ave1</td>
<td>79.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max2+fc_ave2</td>
<td>79.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max3+fc_ave3</td>
<td>79.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max4+fc_ave4</td>
<td>79.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max1+fc_ave1+fc_ave2</td>
<td>79.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max2+fc_ave3+fc_ave4</td>
<td>80.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max1+fc_ave1+fc_ave2+fc_ave3</td>
<td>81.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max1+fc_ave1+fc_ave2+fc_ave3+fc_ave4</td>
<td>81.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>global+local+fc_max1+fc_ave1+fc_ave2+fc_ave3+fc_ave4+fc_ave5</td>
<td>81.1</td>
</tr>
</tbody>
</table>

Table 2
Comparisons of classification rates with other methods on CUB-200.

<table>
<thead>
<tr>
<th>Method</th>
<th>Training Bbox</th>
<th>Part</th>
<th>Testing Bbox</th>
<th>Part</th>
<th>Acc (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DPD [41]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
<td>51.0</td>
</tr>
<tr>
<td>Symbiotic [46]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>61.0</td>
</tr>
<tr>
<td>Alignments [45]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>62.7</td>
</tr>
<tr>
<td>DPD [41]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>64.5</td>
</tr>
<tr>
<td>POOF [43]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>73.3</td>
</tr>
<tr>
<td>Part RCNN [47]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>73.5</td>
</tr>
<tr>
<td>Fully Conv.fct [51]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>75.0</td>
</tr>
<tr>
<td>Pose Normalized CNN [48]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>75.7</td>
</tr>
<tr>
<td>Part-stacked CNN [52]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>76.2</td>
</tr>
<tr>
<td>Part RCNN [47]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>76.7</td>
</tr>
<tr>
<td>Deep LAC [49]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>80.3</td>
</tr>
<tr>
<td>Without Part Annotation</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>82.0</td>
</tr>
<tr>
<td>Without Part Annotation</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>82.8</td>
</tr>
<tr>
<td>Multi-grained [54]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>83.0</td>
</tr>
<tr>
<td>Fully Conv. bilinear [51]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>83.0</td>
</tr>
<tr>
<td>Two-level attention-AlexNet [53]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>69.7</td>
</tr>
<tr>
<td>Two-level attention-VGG-19 [53]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>77.9</td>
</tr>
<tr>
<td>Neural Activation Constellations [55]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>81.0</td>
</tr>
<tr>
<td>Multi-grained [54]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>81.7</td>
</tr>
<tr>
<td>Bilinear [2]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>84.1</td>
</tr>
<tr>
<td>SDN [3]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>84.7</td>
</tr>
<tr>
<td>Ours (fusion)</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>81.8</td>
</tr>
<tr>
<td>Ours (fusion)+Bilinear [2]</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>86.1</td>
</tr>
</tbody>
</table>
best result under the setting of not utilizing bounding box and part annotations, during both training and testing phases. Here, the fusion strategy of our method and bilinear method is late fusing the predictions of the two methods.

4.3. Flower-102 experiments

Flower-102 dataset [61] consists of 102 flower categories with totally 8189 images. We use the standard training-test partition (2040 images for training and 6149 images for testing), which is the same as [61]. As can be seen, the average number of training image for each category is about 20, which means that the problem of Flower-102 recognition is small-sample and to solve this problem seems more challenging. Original dataset also provides the segmentation annotations. But, we do not utilize all these annotations. We first list the results (Table 3) based on our proposed features. Here the evaluated features are fv, global, local, fc_max1 (fc_ave1), fc_max2 (fc_ave2). Only based on these feature representations, we have achieved state-of-the-art performance on this dataset, thus we did not evaluate other representations, such as fc_max3 (fc_ave3) and fc_max4 (fc_ave4).

From Table 3, conclusion can be drawn that 1) global feature is much better than its global baseline (obtained from directly fine-tuning VGG-16), i.e., 94.4% vs. 92.2%; 2) fv feature is also better than its fv baseline (obtained from directly fine-tuning VGG-16), i.e., 96.1% vs. 95.2%; 3) fv representation achieved state-of-the-art performance 96.1% on this dataset; 4) when combined fv with global and local feature, 0.5% improvement has been achieved. We further list other state-of-the-art methods in Table 4, from Table 4 it can be concluded that our result is the best among all the compared counterparts.

4.4. Pets-37 experiments

Pets-37 [62] is a dataset which consists of 37 dog and cat categories. There are 7349 image in total. We use the provided training and test partition, i.e., 3680 images for training and 3669 images for testing. Similar as the previous experiments, we first list the results under our proposed features on LG-CNN. Here the used features are fv, global, local, fc_max1 (fc_ave1), fc_max2 (fc_ave2). Only based on these feature representations, we have achieved state-of-the-art performance on this dataset, thus we did not evaluate other representations, such as fc_max3 (fc_ave3) and fc_max4 (fc_ave4).

From Table 5, it can be seen that the global feature still outperform its baseline feature (91.33% vs. 90.05%), and fv feature outperforms its baseline feature (90.60% vs. 88.87%). Our fusion result has achieved 92.19% accuracy, which is the state-of-the-art on this dataset.

From Table 6, it can be concluded that our fusion result (92.19%) is the current state-of-the-art performance.

4.5. Parameter analysis

In this section, we conduct experiments to evaluate the performance of LG-CNN framework under different parameter settings. We take CUB-200 dataset as an example to carry out all the experiments. Specifically, we first illustrate the influence of the number of part boxes on the accuracy. Secondly, we show the influence of the size of the part boxes on the accuracy. Then we draw the convergent curves of both up stream and bottom stream in LG-CNN. Time consumption of our whole LG-CNN framework is also discussed. We further show the performances of “global” and “local” features w.r.t. different values of λ. Details of feature dimensionality are discussed, and comparisons of linear SVM and soft-max classifier (single-layer perception) are also presented. Furthermore, ablation analysis of each component in LG-CNN is conducted. Finally, we apply the LG-CNN framework to generic object recognition task, and validate that LG-CNN still works well on the generic object recognition task.

4.5.1. The influence of the number of part boxes

In this part, we evaluate the influence of the number of part boxes on the accuracy. We report the accuracies of both global feature (global) and local feature (local) w.r.t. the number of part boxes.
boxes (generated from Fig. 2) on CUB-200. From Fig. 7, it can be concluded that (1) the performance of global feature is much better than the local feature; (2) with the increasing of the part box number, the performance of global feature first becomes better, and then remain stable after using more than 10 boxes; (3) the performance of local feature is always stable (fluctuating within one percentage). In all our experiments, we choose to utilize 25 part boxes, where 5 boxes are the global box and $2 \times 2$ partitioned boxes and the rest 20 boxes are generated based on the unsupervised part localization procedure (in Fig. 2). In these experiments of Fig. 7, the size of the 20 part boxes are fixed as $50 \times 50$, and other parameters for training these LG-CNN are also fixed.

4.5.2. The influence of the size of part boxes

To observe accuracy w.r.t. the size of part boxes, in this part, we fix the part box number (fixed as 25) and other parameters. The accuracies of global and local features under different part box size are drawn (Fig. 8). It can be seen from Fig. 8 that (1) the performance of global feature is stable w.r.t. part box size; (2) with the increasing of part box size, the performance of local feature is becoming better, and then tends to be stable. In all our experiments, the box size is fixed as $50 \times 50$, which can preserve good performance for both global and local features.

4.5.3. Convergence analysis

As LG-CNN has two streams (two loss functions), we draw the convergence curves of LG-CNN on the two streams. We illustrate the convergence curves of LG-CNN training on CUB-200 and Flower-102 respectively. It can be seen from Fig. 9 that the upper stream loss (Global) and bottom stream loss (Local loss) decline rapidly, i.e., LG-CNN converges very fast. In practice, less than 30,000 iterations of LG-CNN is enough to achieve good performances.

4.5.4. Time complexity

Our proposed LG-CNN framework seeks to solve the fine-grained recognition problem without utilizing both bounding box and part annotations. The framework consists of two stages, i.e., (1) unsupervised part and global object discovery; (2) LG-CNN training. The total training time consumption $T$ will be the sum of the time consumptions from these two stages. As for unsupervised part discovery, the total time consumption $T_1$ is less than two hours. The time consumption $T_2$ of global object discovery stage is less than 30 min. As for the LG-CNN training, the time consumption $T_3$ will be longer, about ten hours. Thus, the total time consumption $T = T_1 + T_2 + T_3$ will be more than ten hours. In recent years, CNN based training (fine-tuning) is still very time consuming. Nevertheless, the testing time can be very fast (within seconds) after obtaining the various types of features based on the trained LG-CNN.

4.5.5. Discussion about parameter $\lambda$

In this part, taking CUB-200 as an example, we conduct experiments to observe the changing tendency for the performances of “global” and “local” features w.r.t. the parameter $\lambda$. During the LG-CNN training, the values of $\lambda$ are taken from $[0, 0.1, 0.3, 0.5, 0.7, 0.9, 1.0]$, and other parameters are fixed. As can be seen from Fig. 10, the results of “global” and “local” features are stable w.r.t. different values of $\lambda$. Here, $\lambda = 0$ means that we do not utilize the upper stream loss ($L_u$), thus no result of “local” feature in axis-y (corresponding to $\lambda = 0$) is reported in Fig. 10. In this paper, we set $\lambda = 0.5$ in all the experiments.
CNN features as input [67]; (2) linear SVM has comparable performance as single layer perception classifier; (3) linear SVM is usually used to fuse multiple features for fine-grained recognition (e.g., local and global features) [50]; (4) the direct output of LG-CNN is the average classification accuracy, while the evaluation criteria in the literatures (and our reported results) is mean class-wise accuracy. We further list the comparisons of linear SVM classifier versus direct prediction by soft-max layer outputs (single layer perception) for the “global” and “local” features (Table 8).

4.5.7. Ablation analysis of each component in LG-CNN
LG-CNN consists of (1) unsupervised data-preprocessing and (2) training with weight sharing, which should take the data from (1) as inputs for network training. Furthermore, unsupervised data-preprocessing consists of unsupervised part discovery and global object discovery, wherein th former should be fixed for effective LG-CNN training. Here, we further report the results of “global” and “local” features from LG-CNNs, which are trained by taking various types of images as inputs to the bottom stream of themselves, meanwhile other components of the trained LG-CNNs are fixed. The evaluated key components include: (i) only taking original global images as input to bottom stream of LG-CNN (only original); (ii) only taking the approximate global images (based on saliency detection procedure) as input to bottom stream of LG-CNN (only saliency); (iii) only taking the approximate global images (based on segmentation procedure) as input to bottom stream of LG-CNN (only segmentation); (iv) taking images from (ii) and (iii) as input to bottom stream of LG-CNN (saliency and segmentation); (v) taking images from (i) and (ii) as input to bottom stream of LG-CNN (saliency and original); (vi) taking images from (i) and (iii) as input to bottom stream of LG-CNN (segmentation and original); (vii) taking images from (i), (ii), and (iii) as input to bottom stream of LG-CNN (original, saliency and segmentation); From Table 9, it can be concluded that taking images from saliency and segmentation methods can boost the performances more significantly.

4.5.8. Transfer to generic object recognition
In this part, we take general object recognition dataset, Caltech101 [69], as an example to validate the effectiveness of LG-CNN framework. The objects in the images of Caltech101 usually occupy
Table 10
Comparisons of classification rate (global feature) for LG-CNN and VGG-16-ft on Caltech101.

<table>
<thead>
<tr>
<th>Iterations</th>
<th>LG-CNN (%)</th>
<th>VGG-16-ft (%)</th>
<th>Dimensionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>10k</td>
<td>94.57</td>
<td>93.21</td>
<td>4096</td>
</tr>
<tr>
<td>15k</td>
<td>94.56</td>
<td>93.13</td>
<td>4096</td>
</tr>
<tr>
<td>20k</td>
<td>94.56</td>
<td>93.17</td>
<td>4096</td>
</tr>
<tr>
<td>25k</td>
<td>94.56</td>
<td>93.21</td>
<td>4096</td>
</tr>
<tr>
<td>30k</td>
<td>94.56</td>
<td>93.21</td>
<td>4096</td>
</tr>
</tbody>
</table>

a large region of the images, i.e., the objects in Caltech101 are well cropped ones. Based on the above observation, we do not need to utilize saliency detection and segmentation methods to discover the global objects. As for part candidate localization, the same procedure as Fig. 2 is utilized except that we do not use the aid of saliency detection.

Caltech101 is composed of images belonging to 101 categories and one background category, with totally 9144 images. As for the training-test partition, we use 30 images per category for training and up to 50 images per category for testing. Before CNN training, all the images are first resized to $256 \times 256$. The key parameters for LG-CNN training are the same as the ones in CUB-200 except that the final layer output is changed to 102.

From Table 10, it can be concluded that our LG-CNN framework can still outperform directly fine-tuned VGG-16 net greatly. Moreover, the convergence speed is still very fast on this object recognition dataset, i.e., after 10,000 iterations the best performance has been achieved.

![Fig. 11. Classification rate of each category on CUB-200 dataset.](image)

4.6. Study of failure cases

In this subsection, we study the failure cases of LG-CNN by taking CUB-200 as an example. Specifically, we study the failure cases while taking “fv+global+local +fc_mirror+fc_max1-4” as the input feature. We first show the classification rate of each category on the CUB-200 dataset (Fig. 11). Then we illustrate the image samples from the worst two categories, which have lower classification rate compared with the other categories. Specifically, the worst two categories are california gull and common tern. It can be seen from Fig. 12 that 1) the misclassified categories for almost all the failure cases are their neighbor genus, e.g., six images (california gull) in Fig. 12(a) are misclassified into glaucous winged gull (i.e., 59→60); 2) very few images are misclassified into their non neighbor genus. As can be seen, our proposed LG-CNN framework is mainly focused on input image preprocessing stage, to solve the above misclassifying problems, super categories of the fine-grained images should be considered, which will be our future works.

![Fig. 12. Failure cases of categories: (a) california gull, and (b) common tern. Here 59→62 means that image from category “59” is misclassified into category “62”.](image)
5. Conclusion and future works

In this paper, we have proposed a novel framework to solve the fine-grained recognition problem. We consider a hard problem of not utilizing bounding box and part information during training and testing phase, which is the most difficult case and widely happen in practice. To solve this problem, our system first performs unsupervised part candidates discovery and global object discovery. After that, a two stream CNN network architecture is proposed to model both the local part information and the global discriminative information in a joint network (which is denoted as LG-CNN). In LG-CNN, the weights are shared before the last convolutional layer, the upper stream of LG-CNN is used for capturing local part information by our straighten layer, and the bottom stream of LG-CNN is used for preserving the original global information of input image. Based on the trained LG-CNN, we further construct four types of features, which are combined and fed into the linear SVM for the final prediction. Experiments on three popular fine-grained datasets demonstrate the effectiveness of the proposed fine-grained recognition system. Particularly, we achieved new state-of-the-art results on the CUB-200 dataset, and the results on Flower-102 and Pets-37 are competitive.

In the future, we will consider applying our proposed system to other challenging fine-grained recognition problems. Fisher vector has been a powerful feature representation, to fuse our proposed global/local features with Fisher vector representation, we have to extract various kinds of features from LG-CNN, followed by linear SVM training and prediction; if we can implement Fisher vector into CNN for end-to-end training, then linear SVM can be avoided, thus all the feature fusions presented in this paper can be realized by directly establishing the corresponding CNN architectures. However, it is hard to implement the Fisher vector coding into the CNN architecture, thus realizing feature fusing in an end-to-end way remains a future work.

Acknowledgments

This work has been supported in part by the National Basic Research Program of China (973 Program) Grant 2012CB316302, the Strategic Priority Research Program of the CAS (Grants XDA06040102 and XDB02060009), the Natural Science Foundation of China (NSFC) under Grants 61472370 and 61672469, and the open project of State Key Laboratory of Virtual Reality Technology and System under Grant BUAA-VR-16KF-07.

References


Guo-Sen Xie is currently an Assistant Professor at the Department of Information Engineering at Henan University of Science and Technology. He received the Ph.D. degree in pattern recognition and intelligent systems from Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2016. He received the Best Student Paper Awards from MMM’16. His research interests include machine learning, deep learning, and their applications to object recognition and DNA sequence analysis.

Xu-Yao Zhang received the B.S. degree in computational mathematics from Wuhan University, Wuhan, China, in 2008, and the Ph.D. degree in pattern recognition and intelligent systems from Institute of Automation of Chinese Academy of Sciences, Beijing, China, in 2013. From July 2013, he has been an Assistant Professor at the National Laboratory of Pattern Recognition, Institute of Automation of Chinese Academy of Sciences, Beijing, China. His research interests include machine learning, pattern recognition, handwriting recognition, and deep learning. He has published over 30 papers in international journals and conferences.

Wenhan Yang received the B.S. degree in computer Science from Peking University, Beijing, China, in 2012, where he is currently working toward the Ph.D. degree with the Institute of Computer Science and Technology. His research interests include image and video processing, image interpolation, deep learning-based image restoration and signal super-resolution.

Mingliang Xu is an associate professor in the School of Information Engineering of Zhengzhou University, China, and currently is the director of CIISR (Center for Interdisciplinary Information Science Research), and the general secretary of ACM SIGAI China. His research interests include virtual reality and artificial intelligence. Xu got his Ph.D. degree in computer science and technology from the State Key Lab of CAD&CG at Zhejiang University.

Shuicheng Yan is currently an Associate Professor at the Department of Electrical and Computer Engineering at National University of Singapore, and the founding leader of the Learning and Vision Research Group (http://www.lv-nus.org). Dr. Yan’s research areas include machine learning, computer vision and multimedia. He has authored/co-authored hundreds of technical papers over a wide range of research topics, with Google Scholar citation > 19,000 times and H-index 51. He is ISI Highly-cited Researcher, 2014 and IAPR Fellow 2014. He has been serving as an associate editor of IEEE TKDE, TCSVT and ACM Transactions on Intelligent Systems and Technology (ACM TIST). He received the Best Paper Awards from ACM MM13 (Best Paper and Best Student Paper), ACM MM12 (Best Demo), PCM11, ACM MM10, ICME10 and ICIMCS’09, the runner-up prize of ILSVRC’13, the winner prize of ILSVRC14 detection task, the winner prizes of the classification task in PASCAL VOC 2010–2012, the winner prize of the segmentation task in PASCAL VOC 2012, the honourable mention prize of the detection task in PASCAL VOC10, 2010 TCSVT Best Associate Editor (BAE) Award, 2010 Young Faculty Research Award, 2011 Singapore Young Scientist Award, and 2012 NUS Young Researcher Award.

Cheng-Lin Liu is a Professor at the National Laboratory of Pattern Recognition (NLPR), Institute of Automation of Chinese Academy of Sciences, Beijing, China, and is now the director of the laboratory. He received the B.S. degree in electronic engineering from Wuhan University, Wuhan, China, the M.E. degree in electronic engineering from Beijing Polytechnic University, Beijing, China, the Ph.D. degree in pattern recognition and intelligent control from the Institute of Automation of Chinese Academy of Sciences, Beijing, China, in 1989, 1992 and 1995, respectively. He was a postdoctoral fellow at Korea Advanced Institute of Science and Technology (KAIST) and later at Tokyo University of Agriculture and Technology from March 1996 to March 1999. From 1999 to 2004, he was a research staff member and later a senior researcher at the Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan. His research interests include pattern recognition, image processing, neural networks, machine learning, and especially the applications to character recognition and document analysis. He has published over 200 technical papers at prestigious international journals and conferences. He is on the editorial board of international journals Pattern Recognition, Image and Vision Computing, International Journal on Document Analysis and Recognition, and Computational Geometry. He is a fellow of the IEEE and the IAPR.