AutoBD: Automated Bi-Level Description for Scalable Fine-Grained Visual Categorization

Hantao Yao, Shiliang Zhang, *Member, IEEE*, Chenggang Yan, Yongdong Zhang, *Senior Member, IEEE*, Jintao Li, and Qi Tian, *Fellow, IEEE*

***Abstract*— Compared with traditional image classification, fine-grained visual categorization is a more challenging task, because it targets to classify objects belonging to the same species, *e.g.*, classify hundreds of birds or cars. In the past several years, researchers have made many achievements on this topic. However, most of them are heavily dependent on the artificial annotations, *e.g.,* bounding boxes, part annotations, *and so on*. The requirement of artificial annotations largely hinders the scalability and application. Motivated to release such dependence, this paper proposes a robust and discriminative visual description named Automated Bi-level Description (AutoBD). "Bi-level" denotes two complementary part-level and object-level visual descriptions, respectively. AutoBD is "automated," because it only requires the image-level labels of training images and does not need any annotations for testing images. Compared with the part annotations labeled by the human, the image-level labels can be easily acquired, which thus makes AutoBD suitable for large-scale visual categorization. Specifically, the part-level description is extracted by identifying the local region saliently representing the visual distinctiveness. The object-level description is extracted from object bounding boxes generated with a co-localization algorithm. Although only using the image-level labels, AutoBD outperforms the recent studies on two public benchmark, *i.e.*, classification accuracy achieves 81.6% on CUB-200-2011 and 88.9% on Car-196, respectively. On the large-scale Birdsnap data set, AutoBD achieves the accuracy of 68%, which is currently the best performance to the best of our knowledge.**

Manuscript received July 19, 2016; revised February 6, 2017 and July 17, 2017; accepted September 8, 2017. Date of publication September 13, 2017; date of current version October 17, 2017. This work was supported in part by the National Nature Science Foundation of China under Grant 61525206, Grant 61572050, Grant 91538111, Grant 61429201, and Grant 61428207 and in part by the Beijing Advanced Innovation Center for Imaging Technology under Grant BAICIT-2016009. The work of Q. Tian was supported in part by ARO under Grant W911NF-15-1-0290 and in part by the Faculty Research Gift Awards by NEC Laboratories of America and Blippar. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. David Clausi. *(Corresponding authors: Yongdong Zhang; Qi Tian.)*

H. Yao and Y. Zhang are with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of the Chinese Academy of Sciences, Beijing 100049, China (e-mail: yaohantao@ict.ac.cn; zhyd@ict.ac.cn).

S. Zhang is with the School of Electronic Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: slzhang.jdl@pku.edu.cn).

C. Yan is with the School of Institute of Information and Control, Hangzhou Dianzi University, Hangzhou 541004, China (e-mail: cgyan@hdu.edu.cn).

J. Li is with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: jtli@ict.ac.cn).

Q. Tian is with the Department of Computer Science, The University of Texas at San Antonio, San Antonio, TX 78249-1604 USA (e-mail: qitian@cs.utsa.edu).

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

Digital Object Identifier 10.1109/TIP.2017.2751960

***Index Terms*— Fine-grained visual categorization, convolutional neural network, automated bi-level description.**

I. Introduction

THE large inner-class variances and subtle inter-class distinctions make fine-grained visual categorization a more challenging task than traditional image classification. To accomplish this task, most of existing studies focus on generating robust and discriminative visual descriptions [1]–[11]. Most of the existing descriptions could be classified into two categories: 1) descriptions at the object level, and 2) descriptions at the local part level. Descriptions at the object level are commonly generated by extracting features from object bounding boxes. By filtering the cluttered backgrounds and effectively depicting the whole foreground object, object-level descriptions show reasonably good performance. Descriptions at the local part level are motivated by the fact that subtle differences among objects commonly located in local regions. For example, as shown in Fig. 1, different species of birds cannot be classified by their global shapes, but can be easily distinguished by local parts like faces. More detailed review of the related studies will be summarized in Sec. II.

By generating better visual descriptions, the existing studies [1], [2], [6] have largely improved the classification accuracy of fine-grained visual categorizations, *e.g.*, pushing the accuracy on CUB-200–2011 [12] from 17.31% [12] to 85.4% [2]. However, one major limitation of these studies is requiring accurate object bounding boxes or local part annotations to generate visual descriptions for both training and testing. Because such annotations are expensive to collect, most of the existing studies are still not mature enough for real applications. Especially for the large-scale categorization tasks, annotating the parts or bounding boxes for millions of images from thousands of categories is too expensive to accomplish.

The goal of this paper is to release the dependency on any artificial annotations, and generate the desired visual descriptions only with image-level labels. A scalable fine-grained visual categorization system can hence be established because the image-level labels for large-scale data are relatively easy to acquire from the Internet. With image-level labels, we generate the Automated Bi-level Description (AutoBD). AutoBD is composed of two complimentary descriptions: the part-level description and object-level description, respectively. As illustrated in Fig. 2, the part-level description is extracted from a region saliently representing the visual distinctiveness, and

Red faced Cormorant Pelagic Cormorant

Fig. 1. Illustration of the subtle difference between two species of bird.

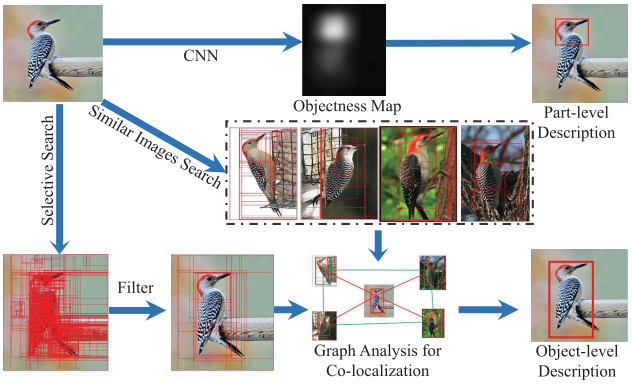


Fig. 2. The illustration of our framework to generate AutoBD.

the object-level description is extracted from the automatically generated object bounding boxes. Given an image, we first apply the Selective Search [13] to generate thousands of candidate bounding boxes. An objectness map is then generated with Convolutional Neural Network (CNN) to estimate the presence of the object on the image. With the help of objectness map, we effectively filter candidate bounding boxes and select the distinctive local regions for part-level description. To estimate object bounding boxes for object-level description, we propose a graph analysis algorithm. The key idea is to co-localize the object using images containing similar objects. Once the two descriptions are computed, we integrate them to obtain the final AutoBD description.

There exist some studies that also generate visual descriptions only with image-level label [4], [5], [14], [15]. However, we are different with them in several aspects. In [15], Lin *et al.* use a bilinear CNN to generate the description for the whole image. Similarly, Ge *et al.* [14] learn the CNN model and generate the final description from the whole image. Xiao *et al.* [5] select the object parts simply by a trained DomainNet [5], and Simon and Rodner [4] apply a constellation of neural activation patterns to generate the object parts. Compared with them, AutoBD extracts both object-level and part-level descriptions in an efficient framework. Experimental results also manifest our advantages over them.

To validate our work, we evaluate the proposed AutoBD on two popular fine-grained visual categorization datasets: CUB-200-2011 [12] and Car-196 [16]. Note that, we only use image-level annotation for training and do not use any annotation for testing. Experiments show that our work outperforms existing studies, *i.e.*, we achieve the classification accuracy of 81.6% on CUB-200-2011 [12], 88.9% on Car-196 [16]. Because our AutoBD only requires image-level labels for training, we manage to test it on a large-scale classification task on Birdsnap dataset [17], which contains 49,828 images from 500 categories. We achieve the accuracy of 68%. We could hence summarize our contributions as followings:

1) We propose two effective algorithms to automatically extract the object-level and part-level representations, respectively.
2) Only with the image-level labels, AutoBD performs impressively on both the small and the large-scale datasets. This clearly demonstrates the robustness, discriminative power, and scalability of AutoBD.
3) To the best of our knowledge, this is one of a few studies studying large-scale fine-grained visual categorization, which is more challenging and valuable for real scenarios. The success of this work guarantees further investigation on this task.

The remainder of this paper is organized as follows. Sec. II reviews the related work. Sec. III presents the detailed descriptions for AutoBD. Sec. IV introduces our experiments, followed by the conclusions in Sec. V.

II. Related Work

In this section, we will briefly review the related methods for fine-grained visual categorization. During the past five years, fine-grained visual categorization has attracted lots of attention from both the academic and industrial communities. Most of existing studies require bounding boxes or human labeled part annotations to construct discriminative visual description [1]–[10], [18]–[22]. According to the cues used in generating visual description, we summarize these studies into two categories, *i.e.*, studies on local part learning and object localization, respectively.

Local part learning: As shown in Fig. 1, some distinctive clues of objects commonly exist in their local parts. Thus, generating visual descriptions from local parts is helpful to achieve high discriminative power. Motivated by this, various studies have been proposed to utilize or learn local parts. The early studies, *e.g.*, Xie *et al.* [19] and Berg and Belhumeur [1], directly employ the ground truth part annotations for both training and testing. When the part annotations for testing images are unknown, several studies infer such cues from part annotations of training images. For example, Goering *et al.* [23] and Gavves *et al.* [18] apply global and local similarities to learn the part annotations for testing images. Liu and Belhumeur [9] infer parts from some similar exemplars using the pose and subcategory consistency. Given a test image, Branson *et al.* [2] first infer keypoints, then employ keypoints to align the generated parts.

Local parts can also be generated by selecting meaningful candidate bounding boxes. Zhang *et al.* [6] apply R-CNN [24] and geometric constraints to select distinctive candidate

bounding boxes as parts. Xiao *et al.* [5] apply the part detector generated by spectral clustering to select parts from candidate bounding boxes. Recently, Lin *et al.* [3] propose an integrity Deep LAC model to localize and align parts. This model contains a localization sub-network, an alignment sub-network, and a classification sub-network, respectively. As the *conv* or *pool* feature maps in CNN convey the location cues, several studies [8], [22] employ the *conv* or *pool* feature maps to infer part annotations. Souri and Kasaei [22] first employ a pretrained CNN to extract the *conv* feature maps, then train a random forest classifier [25] to classify the pixels into parts. Different from [22], Zhang *et al.* [8] train an end-to-end fully convolutional networks [26] to infer the part annotations for testing images.

Object Localization: Object bounding boxes discard the cluttered backgrounds and depict the global visual property. They also play a significant role in fine-grained object description. To obtain the object bounding box, two popular methods are Deformable Part Model (DPM) [27] and R-CNN [24], respectively. Zhang *et al.* [7] and Chai *et al.* [21] both employ DPM for object detection. Zhang *et al.* [6] apply the R-CNN [24] to generate the bounding boxes. Because it is expensive to collect bounding box annotations, unsupervised object discovery only using image-level labels is more attractive. In fine-grained visual categorization, two recent studies [4], [5] have generated the object regions only with the image-level labels.

Feature Description: Once the part-level and object-level regions are generated, how to describe their visual contents is another key issue. In traditional methods, Histogram-Of-Gradient (HOG) [28], Bag-Of-visual-Word (BOW) [29], Fisher Vector [30] and kernel descriptors (KDES) [31] have been widely utilized. Among these features, Fisher Vector achieves the state-of-the-art performance for fine-grained object description [18].

Currently, due to the impressive performance of CNN [32] on large scale visual recognition, more and more studies use CNN as feature extractor. Among these studies, Branson *et al.* [2], Zhang *et al.* [6], Lin *et al.* [3], Simon and Rodner [4], and Xiao *et al.* [5] apply CNN features for fine-grained visual categorization. These studies extract features through a CNN which is first trained on the ImageNet [33] and then fine-tuned on the target dataset. In their experiments, CNN features show a 10% improvement over the traditional features. Inspired by this, we apply the CAFFE [34], which is an open source deep CNN toolbox for fine-tuning and extracting features.

This is work is motivated by how to generate the desired visual descriptions only with image-level label. Among existing studies, the most related one to ours is [5], which describes the objects from two-level attention parts. Our work differs from it in the following aspects: 1) for object-level description, we propose a graph analysis algorithm to co-localize the object. This is more accurate than [5], which simply applies a trained DomainNet [5] to classify the candidate object regions; 2) we generate the local regions for part-level description with objectness map. This also differs from [5], which trains part detectors with spectral clustering. Extensive experiments also show that our algorithm obtains significantly higher accuracy than [5].

III. Automated Bi-Level Description

A. Problem Formulation

Given n images (x_i, y_i), where x_i and y_i are the i-th image and its label, the fine-grained visual categorization problem could be formulated as Eq. (1),

$$\min_{\boldsymbol{\omega}} \frac{1}{2}\boldsymbol{\omega}^\top \boldsymbol{\omega} + \mathcal{C}\sum_{i=1}^{n} \epsilon(\boldsymbol{\omega}; \boldsymbol{f_i}, y_i), \tag{1}$$

where $\boldsymbol{f_i}$ is the description for image x_i, and $\boldsymbol{\omega}$ is the parameter for classifier. For convenience, we simplify the $\boldsymbol{f_i}$ as $\boldsymbol{f}$ in this work.

As $\boldsymbol{\omega}$ could be inferred by various optimization methods [35], the key of fine-grained visual categorization is how to generate a robust descriptor $\boldsymbol{f}$. To exclude cluttered backgrounds and ensure its discriminative power to different levels of visual cues, a powerful $\boldsymbol{f}$ could consist of two components: object-level description $\boldsymbol{f^b}$ and part-level description $\boldsymbol{f^p}$, respectively,

$$\boldsymbol{f} = [\boldsymbol{f^b}, \boldsymbol{f^p}], \tag{2}$$

where $\boldsymbol{f^b}$ depicts the object bounding box and $\boldsymbol{f^p}$ focuses on the discriminative object parts.

Most of existing methods [4], [5], [15] focus on how to generate a image-level description or part-level description, and pay less attention on the object-level description $\boldsymbol{f^b}$. For example, Lin *et al.* [15] propose a Bilinear-CNN to generate robust image description, and Xiao *et al.* [5] and Simon and Rodner [4] improve the classification accuracy by generating powerful part-level description. These descriptions either suffer from low discriminative power or expensive computation and heavy dependence on part annotations. These motivate us to study: 1) unsupervised object localization method for object-level description generation, and 2) more effective part-level description generation algorithm that does not need any part annotation. We called the proposed methods as Automated Bi-level Description (AutoBD).

Therefore, the AutoBD contains an object-level description and a part-level description. As shown in Fig. 2, we first apply the Selective Search [13] to generate thousands of candidate bounding boxes. The objectness map, which is generated by CNN, is then used to filter noisy bounding boxes and select the most discriminative regions for part-level description. Finally, we propose a graph analysis co-localization algorithm for object-level description. In the following parts, we introduce how to generate the objectness map, filtering the noisy bounding boxes, and generate part-level and object-level descriptions, respectively.

B. Objectness Map Generation

We define the objectness map as an image where each pixel value indicates the probability that the pixel belongs to an interested object. As the Convolutional Neural Network (CNN) [32] has exhibited strong discriminative power

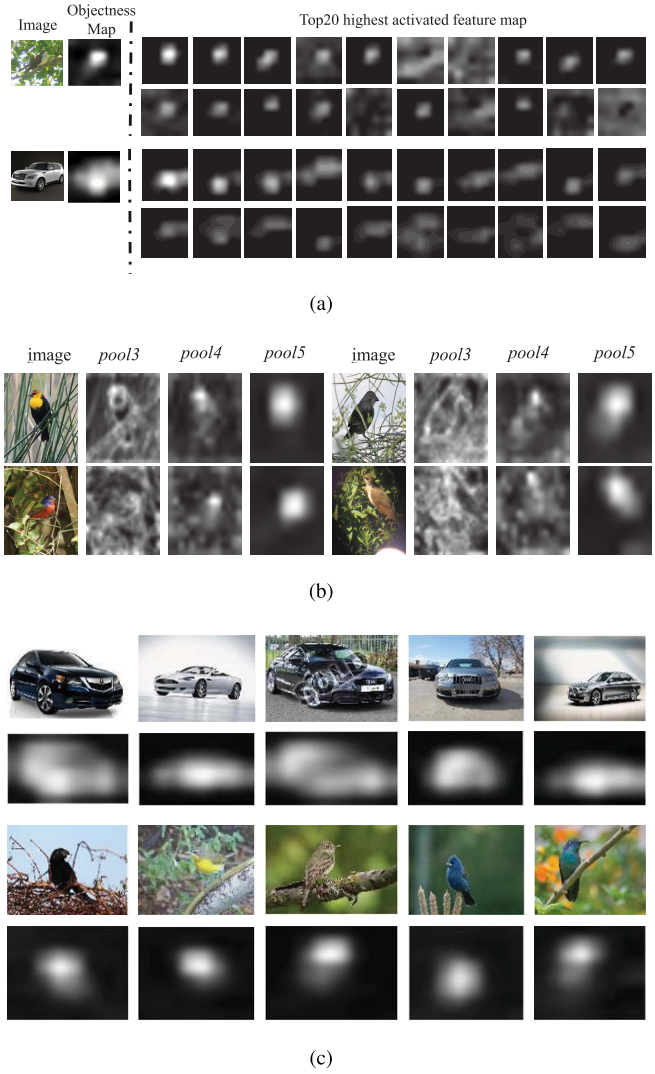


Fig. 3. (a) Illustration of objectness map and feature map. (b) Objectness maps generated by different layers of *pool* feature maps. (c) More examples of generated objectness map for cars and birds.

for object segmentation and classification, we use the feature maps in CNN to estimate the presence of objects. We find that applying average-pooling on *conv* or *pool* feature maps obtains an image reasonably representing the object location. For example, as shown in Fig. 3 (a), although there exist some feature map with high activation on the background, most of feature maps focus on the foreground object region. As a consequence, applying average pooling on those feature maps would suppress the activation on background, meanwhile gather and emphasize the activation on foreground region.

To generate the objectness map, we fine-tune a pre-trained CNN, *i.e.*, VGG-19 [36], in a classification task. The goal of this training is to learn convolutional filters that are discriminative to the interesting objects like bird, dog, car, aircraft, *etc*. The detailed description of the training set is introduced in Sect. IV-A. Once this CNN is fine-tuned, we extract its *pool* feature maps to generate objectness map. Note that, the *pool* feature represents the responses of different convolutional filters on the object. With enough discriminative power, the filters will be activated when they "see" their identifiable objects, and will not respond otherwise. Therefore, the feature values of *pool* necessarily indicate the presence of an object.

As there are multiple *pool* layers in VGG-19 net, we analyze the objectness map generated by different layers, *e.g., pool3, pool4, pool5*. The objectness maps generated by different layers are illustrated in Fig. 3 (b). Compared with the objectness map generated by *pool5* layer, the objectness maps generated by *pool3* and *pool4* have higher response on the background regions. This is because, compared with *pool3* and *pool4*, *pool5* is trained to generated a higher-level description closely related with the classification task, which needs to focus on the foreground region and filter the background region. Therefore, it is more rational to use *pool5* to generate the objectness map.

To generate the objectness map for a given image I, we first resize it to size 256×256, then feed it into the CNN to extract its *pool*5 feature, denoted as f_I^{p5}. For example, the size of f_I^{p5} is $512 \times 8 \times 8$ for VGG-19 net, where the 512 is the number of convolutional filters in $conv5$ layer in VGG-19. With the *pool*5 feature, the objectness map m_I is thus generated by applying average pooling on f_I^{p5} with Eq. (3), *i.e.*,

$$m_I(i, j) = \sum_{c=1}^{C} f_I^{p5}(c, i, j), \tag{3}$$

where $C = 512$ for *pool*5 layer in VGG-19. We then normalize m_I by dividing its max, and resize it to the same size of image I. Fig. 3 (c) shows more objectness maps for birds and cars. It is obvious that, objectness maps clearly indicate the rough locations of objects, hence can be utilized to filter the noisy candidate object bounding boxes.

C. Bounding Boxes Filtering

Selective Search generates thousands of candidate bounding boxes. We use the objectness map to filter the noisy ones. Given an image I, we denote its objectness map as m_I, and its bounding boxes generated by Selective Search as $\mathcal{B}_I^{ss}$. For the objectness map m_I, we employ a threshold θ to turn it into a binary image. Specifically, we consider a pixel as the foreground pixel if its value is larger than θ. As shown in Fig. 4, for the binary image, we treat its minimum enclosing rectangle as a hypothetical bounding box, denoted as b^h, *e.g.,* the green bounding boxes are the hypothetical boxes for each binary image in Fig. 4. For each candidate bounding box b ($b \in \mathcal{B}_I^{ss}$), we compute its overlap score with hypothetical bounding box b^h. Specifically, we use $precScore(b, b^h)$ to measure the overlap score referring to [37], *i.e.,*

$$precScore(b, b^h) = \frac{area(b \cap b^h)}{area(b \cup b^h)}, \tag{4}$$

where $b \cap b^h$ denotes the intersection of the bounding boxes b and b^h, and $b \cup b^h$ is their union.

For each binary image generated with θ, we select the bounding boxes with top-10 $precScore(\cdot)$scores and discard the rests. In our implementation, because we have no prior knowledge about the optimal threshold θ, we use multiple thresholds to generate the binary images. Five thresholds are used in our work: $[0.1, 0.2, 0.3, 0.4, 0.5]$.

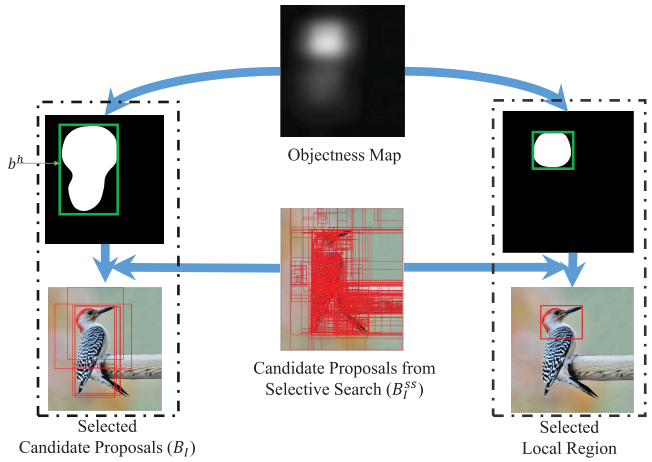


Fig. 4. Illustration of candidate bounding boxes filtering and the generation of local region for part-level description.

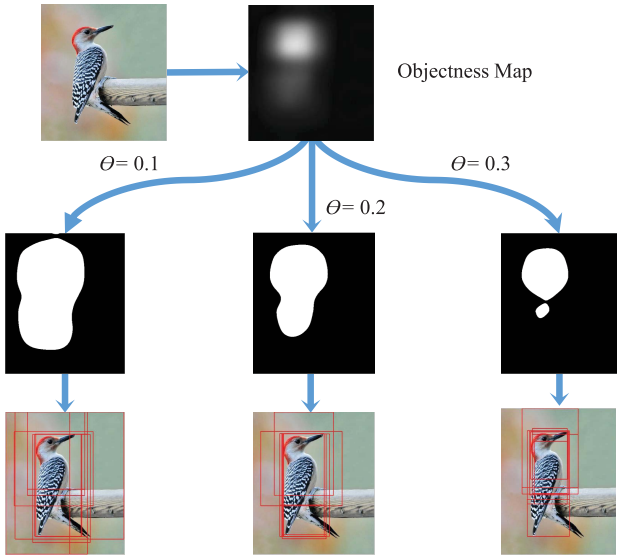


Fig. 5. The effects of multiple thresholds for bounding boxes filtering. This figure show the top-10 selected candidate bounding boxes for three thresholds: 0.1, 0.2, and 0.3, respectively.

The detailed discussion for the parameter θ is presented in Sect. IV-C. Examples of selected bounding boxes are shown in Fig. 5 and Fig. 13. Because we select 10 bounding boxes for each θ, we finally obtain about 50 bounding boxes for each image. After deleting some duplicate ones, there roughly exist 30 bounding boxes finally. For the image I, we denote its filtered bounding boxes as $\mathcal{B}_I$.

D. Object-Level Description

With the filtered bounding boxes, we propose a graph analysis algorithm to estimate the accurate object bounding boxes, which is hence used to generate the object-level description.

Given a test image I along with its bounding boxes $\mathcal{B}_I$, we first retrieve its K Nearest Neighbor (KNN) images from all training images. We denote the KNN images as $\{N_1, N_2, \ldots, N_K\}$, and their candidate bounding boxes as $\{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_K\}$. The KNN images are searched with the $fc6$ feature from the fine-tuned CNN model using *cosine* distance. The parameter K will be discussed in Sect. IV-D.

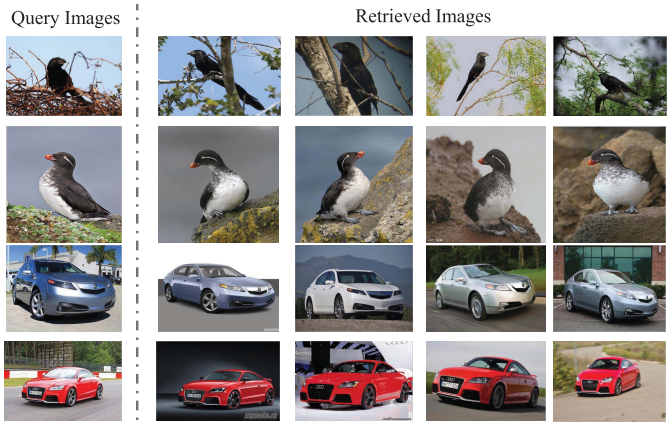


Fig. 6. Examples of retrieved KNN images with the $fc6$ feature.

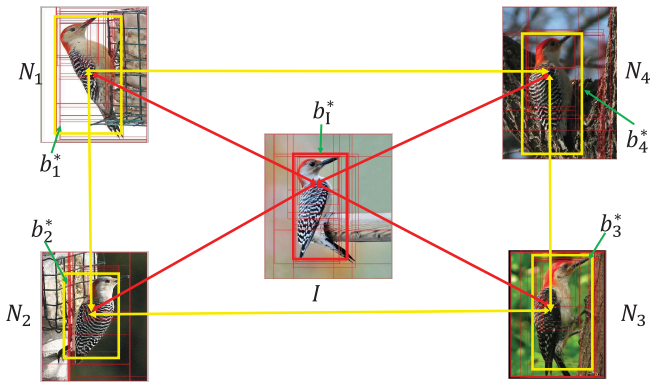


Fig. 7. The illustration of graph-based object co-localization. The graph consists of one test image I and 4 Nearest Neighbor images. The red bounding box b_I^* is the selected object bounding box for image I. $b_k^*(k \in [1, 2, 3, 4])$ are the nearest bounding boxes of b_I^* in the 4 Nearest Neighbor images.

As shown in Fig. 3, the CNN activations on objects are substantially higher than the ones on backgrounds. It could be inferred that, CNN feature implicitly filters backgrounds and focuses on the objects. Therefore, as shown in Fig. 6, the retrieved KNN images would contain similar object and diverse backgrounds. Accordingly, their accurate object bounding boxes would share strong visual consistency. For example, the similarities among the bounding boxes on birds in Fig. 7 would be higher than the similarities among the bounding boxes covering some backgrounds. Motivated by this, we construct a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with $\mathcal{V} = \{\mathcal{B}_I, \mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_K\}$, where $\mathcal{B}_I$ is the set of selected bounding boxes for testing image I, and $\mathcal{E}$ is the set of edges linking bounding boxes among different images. We denote the weight of the edge as $w(\cdot)$ and it reflects the visual distinction between two bounding boxes.

Based on the graph $\mathcal{G}$, we expect to find the most accurate bounding box b_I^* for image I and its corresponding nearest bounding boxes b_k^* for image $N_k(k \in [1, K])$, as illustrated

in Fig. 7. We formulate the selection as:

$$b_I^* = arg \min_{b_I \in \mathcal{B}_I} D_{TN}(b_I) + D_{NN}(b_I), \tag{5}$$

where $D_{TN}(b_I)$ is the sum of distance between target bounding box b_I and its nearest bounding box in each KNN image, *i.e.*,

$$D_{TN}(b_I) = \sum_{k=1}^{K} w(b_I, b_k^*), \tag{6}$$

with

$$b_k^* = arg \min_{b_k \in \mathcal{B}_k} w(b_I, b_k). \tag{7}$$

where $w(\cdot)$ measures the distance between two bounding boxes. b_k^* is the most similar bounding box for target bounding box b_I in $\mathcal{B}_k$.

In Eq. (5), $D_{NN}(b_I)$ is the sum of distance among all nearest bounding boxes of b_I. It is computed as:

$$D_{NN}(b_I) = \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} w(b_i^*, b_j^*). \tag{8}$$

As illustrated in Fig. 7, D_{TN} sums up the weights of red edges and D_{NN} sums up the weights of yellow edges. According to Eq. (5) to Eq. (8), we try to find a sub-graph illustrated in Fig. 7, that has the minimum sum of weights on edges to accurately locate the object.

To make the above mentioned model work, the distance $w(\cdot)$ should be properly defined. We complementary consider the visual distance and the objectness of bounding boxes to define it, *i.e.*,

$$w(b_I, b_k) = \frac{dis(b_I, b_k)}{objScore(b_I) \times objScore(b_k)}, \tag{9}$$

where $dis(\cdot)$ is the *cosine* distance between the visual features of two bounding boxes. In this work, we use the output of $fc6$ layer in CNN as the visual feature.

The reason that we combine the $dis(\cdot)$ and $objScore(\cdot)$ could be summarized into two aspects: 1) $dis(\cdot)$ describes the visual similarity between two bounding boxes described by CNN feature. As shown in Fig. 3, CNN feature implicitly filters backgrounds and focuses on the discriminative regions. This may lead to, for instance, higher similarity between a large bounding box containing lots of backgrounds and another small bounding box containing only discriminative region. 2) $objScore(\cdot)$ describes the probability that one bounding box containing an object. We employ $objScore(\cdot)$ to encourage each bounding box should have a proper size, *i.e.,* tightly contains the object.

We use $objScore(\cdot)$ to estimate the probability that a bounding box contains an object. It is computed by combining two cues: the probability predicted by CNN and the objectness map. It is calculated by Eq.(10), *i.e.*,

$$objScore(b_I) = probScore(b_I) \times mapScore(b_I), \tag{10}$$

where $probScore(\cdot)$ is the probability predicted by the *prob* layer of CNN. Specifically, when image region b_I is input into the CNN, the *prob* layer predicts the probabilities of several objects it can be classified as. We denote the maximum probability as $probScore(b_I)$, *i.e.*,

$$probScore(b_I) = \max(f^{prob}(b_I)), \tag{11}$$

where $f^{prob}(b_I)$ is the output of *prob* layer of CNN. A higher $probScore(b_I)$ means a higher probability that bounding box b_I contains an object.

Because contextual cues on backgrounds may be helpful for classification, bounding boxes with larger $probScore(\cdot)$ commonly contain some backgrounds. Examples are illustrated in Fig. 14(a). We thus also consider $mapScore(\cdot)$ to eliminate the backgrounds.

$mapScore(\cdot)$ is computed based on the objectness map. We denote the obejctness map for bounding box b_I as m_{b_I}, which is computed by cropping the objectness map m_I of I with the region of b_I. The $mapScore(b_I)$ is then calculated with Eq.(12), *i.e.*,

$$mapScore(b_I) = \frac{\sum_{r=1:\mathcal{R},c=1:\mathcal{C}} m_{b_I}(r,c)}{\mathcal{R} \times \mathcal{C}}, \tag{12}$$

where $\mathcal{C}$ and $\mathcal{R}$ are numbers of columns and rows in b_I. It can be seen that $mapScore(\cdot)$ computes the "denseness" of objectness in the bounding box. From Fig. 3, we could see that the local parts could have larger $mapScore(\cdot)$ than the entire object region. Therefore, bounding boxes with large $mapScore(\cdot)$ tend to be smaller than the actual object. Some examples are illustrated in Fig. 14(b).

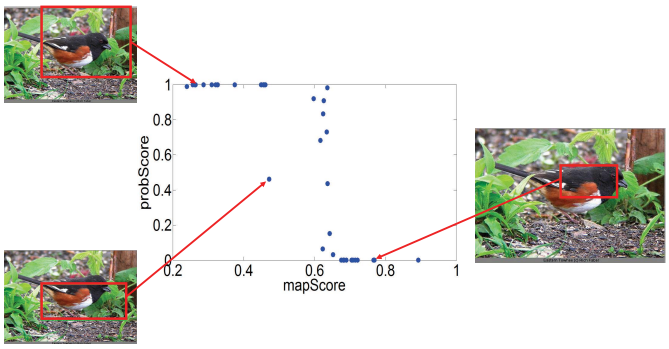


Fig. 8. Three bounding boxes with different sizes and their $probScore(\cdot)$ and $mapScore(\cdot)$, respectively. The more samples are shown in Fig. 14(a) and Fig. 14(b).

As shown in Fig. 8, $probScore(\cdot)$ and $mapScore(\cdot)$ show different properties. The larger bounding box has higher $probScore(\cdot)$ and lower $mapScore(\cdot)$, while the smaller bounding box has higher $mapScore(\cdot)$ and lower $probScore(\cdot)$. Fig. 9 further illustrates this finding. It can be observed that, higher $probScore(\cdot)$ tends to select larger bounding boxes, and higher $mapScore(\cdot)$ selects smaller bounding boxes, respectively. As a consequence, we combine those two cues as the $objScore(\cdot)$ in Eq. (10). As shown in Fig. 14(c), combining these two cues selects more accurate object bounding boxes.

We extract deep features from the selected bounding boxes as the object-level description. In order to improve the discriminative power, we select the top-5 bounding boxes with

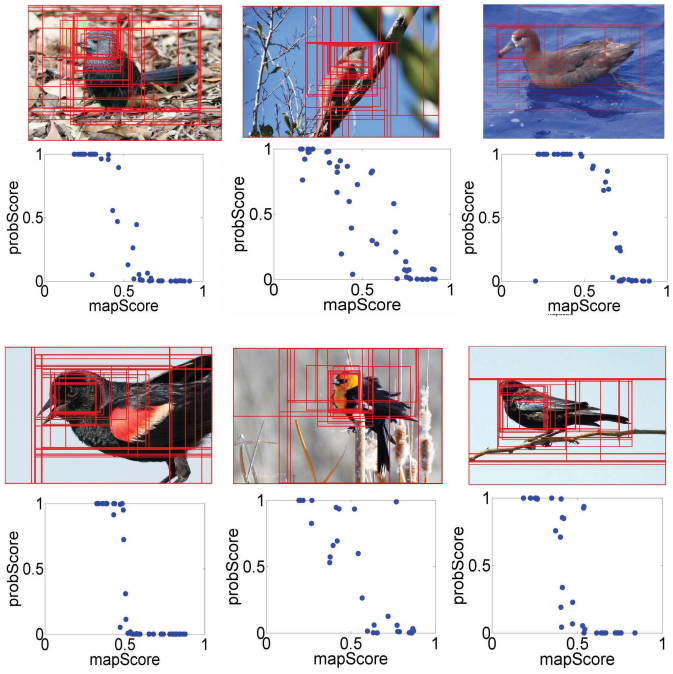


Fig. 9. Illustration of the impact of $probScore(\cdot)$ and $mapScore(\cdot)$ on bounding box size. For each example, the first row shows the source image along with its candidate bounding boxes, and the second row show the $probScore(\cdot)$ and $mapScore(\cdot)$ for those bounding boxes.

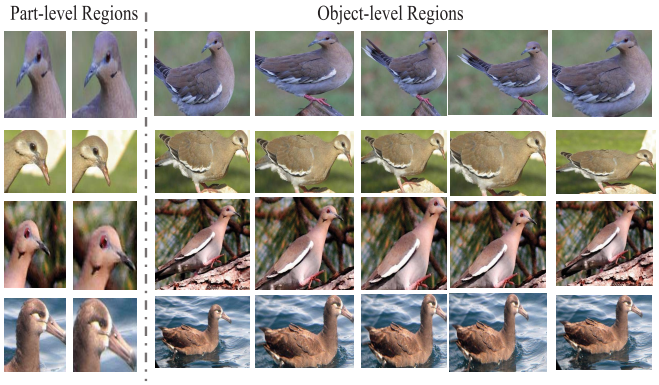


Fig. 10. Examples of the extracted object-level regions and part-level regions.

the highest score according to Eq. (5). The selected bounding boxes denoted as $R^o = \{R_1, R_2, R_3, R_4, R_5\}$. Some samples of the selected bounding boxes are shown in Fig. 10.

E. Part-Level Description

The object-level description lacks the ability to describe the subtle differences, *e.g.,* the heads of birds. Targeting to solve this problem, the part-level description is proposed as another complementary description. We employ the objectness map to select the discriminative part.

Given an image I along with its objectness map m_I, and candidate bounding boxes $\mathcal{B}_I^{ss}$. We utilize the objectness map m_I to choose small and discriminative bounding boxes from $\mathcal{B}_I^{ss}$ as parts. As shown in Fig. 4, when generating the binary images from objectness map, higher threshold corresponds to smaller but more confident and distinctive regions in the object. Therefore, we employ two higher thresholds, *i.e.,* experimentally set as 0.5 and 0.6, to generate the binary images and their corresponding hypothetical bounding boxes.

The hypothetical bounding boxes guide the local part selection in similar way of object bounding box selection. Because the hypothetical bounding boxes b^h can be small because of the higher thresholds. To avoid selecting too small regions, we consider the extra *recall score* constraint. Specifically for each candidate bounding box b ($b \in \mathcal{B}_I^{ss}$), we calculate its score $partScore(b)$ by Eq. (13),

$$partScore(b) = precScore(b) \times recallScore(b), \quad (13)$$

with

$$recallScore(b, b^h) = \frac{area(b \cap b^h)}{area(b^h)}, \quad (14)$$

where $precScore(b)$ is computed in Eq. (4).

From the candidate bounding boxes of each image, we finally select the bounding boxes with the highest $partScore(\cdot)$ as the local regions for part-level description. The selected bounding boxes are denoted as $R^p = \{R_1, R_2\}$, where R_1 and R_2 are the bounding boxes selected by two thresholds 0.5 and 0.6, respectively. Some samples of selected local regions are shown in Fig. 10.

F. Final Bi-Level Description

Once obtaining the part-level and object-level regions R^p and R^o, we employ those image regions to generate the final description. We denote the descriptions for part-level and object-level as $\boldsymbol{f}^p$ and $\boldsymbol{f}^o$, respectively.

For the part-level description $\boldsymbol{f}^p$, we first extract deep feature for each region in $R^p = \{R_1, R_2\}$. We use the $\boldsymbol{f}_i^r$ to denote the feature for region R_i. $\boldsymbol{f}_i^r$ is computed as:

$$\boldsymbol{f}_i^r = \phi(R_i), i = 1, 2, \quad (15)$$

where $\phi(\cdot)$ is the feature extractor.

We then apply *max-pooling* for the local features to generate the final feature $\boldsymbol{f}^p$,

$$\boldsymbol{f}^p(d) = \max_{i \in [1,2]} \boldsymbol{f}_i^r(d), \quad (16)$$

where $\boldsymbol{f}^p(d)$ and $\boldsymbol{f}_i^r(d)$ denote the d-th dimension for $\boldsymbol{f}^p$ and $\boldsymbol{f}_i^r$, respectively.

The the object-level description f^b is computed in similar ways shown above. We finally concatenate $\boldsymbol{f}^p$ and $\boldsymbol{f}^b$ to obtain the final AutoBD description $\boldsymbol{f}$, *i.e.*,

$$\boldsymbol{f} = [\boldsymbol{f}^b, \boldsymbol{f}^p]^{\mathrm{T}}. \quad (17)$$

IV. Experiments

A. Datasets

We evaluate AutoBD on three fine-grained object classification datasets: CUB-200-2011 [12], Car-196 [16], and Birdsnap [17]. CUB-200-2011 is a widely used fine-grained classification dataset, which contains 5,994 training images, and 5,774 testing images. Those images span 200 species, and each contains about 30 training and testing images.

TABLE I
THE IMAGES USED FOR TRAINING CATEGORY-BASED CNN

Category	Dataset	Training Images
Bird	CUB-200-2011 [12]	5,994
Dog	StanfordDogs [38]	12,000
Car	Car-196 [16]	8,144
Aircraft	FGVC-Aircraft [39]	6,667

The Car-196 consists of 16,185 images from 196 classes. The 16,185 images are split into 8,144 training images and 8,041 testing images. Finally, we evaluate our AutoBD on the large-scale Birdsnap dataset. It is also a bird classification dataset, but contains more species and is more challenging than the CUB-200-2011. Birdsnap contains 49,829 images of 500 most common species of birds in North American. Each species contains 60 to 100 images. For all of these datasets, we only use their image-level labels for training.

We also train a category-based CNN to generate the objectness map. The training set contains 32,805 images from four categories, *i.e.* bird, dog, car, and aircraft. The training images are collected from four fine-grained object classification datasets. The detailed information is summarized in Table I.

B. Implementation Details

Two CNNs are trained in our work, *i.e.*, the category CNN for objectness computation and the CNN for feature extraction. We refer to the architecture of VGG-19 [36][1] to build them. The two CNNs are initialized with the model pre-trained on ImageNet [33], and then are fine-tuned on the target datasets. During the fine-tuning, we set the start *learning rate* as 0.001, the *gamma* as 0.1, and the *stepsize* as 5,000 for CUB-200–2011 and Car-196. For the Bridsnap dataset, the *stepsize* is set as 20 000, because this dataset contains a larger number of images.

To extract deep feature from an image region, we first resize the image and then feed it into CNN to extract the *pool5* feature. The size of *pool5* is $512 \times 7 \times 7$. We then apply *max pooling* on each 7×7 feature map, which finally generates the $512D$ deep feature f^{mp5}. After generating the final AutoBD description f, we first apply power- [40] and l_2 normalization, then train a linear SVM [35] as the classifier.

C. Impact of Parameter θ

We set parameter θ to filter the bounding boxes in Sec. III-C. In this section, we analyze and show the impact of it in Table II and Table III based on the training images of each dataset. In Table II, the *Avg. Bbnum* denotes the average number of selected bounding boxes, and the *DR* represents the Detection Rate computed with the metric in [41]. Higher DR means more accurate bounding boxes are selected.

We first show the DR of the raw bounding boxes generated by Selective Search. It can be observed that, Selective Search generates a large number of bounding boxes on CUB-200-2011, thus reasonably achieves a high detection rate of 99.1%.

[1]https://gist.github.com/ksimonyan/3785162f95cd2d5fee77#filereadme-md

TABLE II
THE IMPACT OF PARAMETER θ FOR BOUNDING BOXES SELECTION ON CUB-200-2011

Parameter θ	*Avg. Bbnum*	*DR*(%)
0.1	10	65.83
0.1,0.2	15	89.64
0.1,0.2,0.3	20	95.45
0.1,0.2,0.3,0.4	23	97.32
0.1,0.2,0.3,0.4,0.5	27	97.90
0.1,0.2,0.3,0.4,0.5,0.6	31	98.0
Selective Search	1,672	**99.1**

TABLE III
THE IMPACT OF PARAMETER θ FOR BOUNDING BOXES SELECTION ON CAR-196

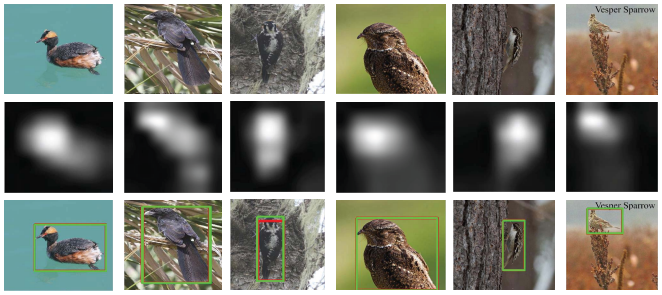
Parameter θ	*Avg. Bbnum*	*DR*(%)
0.1	10	93
0.1,0.2	13	96.24
0.1,0.2,0.3	15	97.39
0.1,0.2,0.3,0.4	18	99.72
0.1,0.2,0.3,0.4,0.5	21	99.86
0.1,0.2,0.3,0.4,0.5,0.6	25	99.94
Selective Search	1,631	**1**

Then, we show the DRs of the filtered bounding boxes by Eq. (4). Note that, each θ selects 10 bounding boxes and the duplicated ones selected by multiple θ will be removed. As shown in Table II, using six θ finally selects about 31 bounding boxes. Table II also shows that, using more θ constantly results in higher DR. For example, using five θ obtains the DR of 97.9%. This means that our algorithm effectively selects accurate ones from the raw bounding boxes produced by Selective Search.

Table III shows the impact of θ for bounding boxes selection on Car-196. From the Table III, we could see that Car-196 has higher DR than CUB-200-2011 under the same setting of θ. The reason might because: 1) each image in Car-196 only contains one object, and 2) the car commonly occupies the majority of the image on Car-196. From Table II and Table III, we could observe that our algorithm is not sensitive to θ, *e.g.,* $\theta = [0.1, 0.2, 0.3, 0.4, 0.5, 0.6]$ obtain similar detection rate with $\theta = [0.1, 0.2, 0.3]$. Therefore, we set $\theta = [0.1, 0.2, 0.3, 0.4, 0.5]$ in the following experiments.

To intuitively show the performance of our bounding box selection, we show some examples of good and bad cases in Fig. 11 and Fig. 13. Fig. 11(a) shows that our algorithm performs well even when the object is small and the background is cluttered. Fig. 11(b) shows several badly cases. The reasons why our algorithm performs bad in Fig. 11(b) could be summarized into two aspects:

1) Some images contain multiple objects, but the ground truth bounding box only covers one object. As shown in Fig. 11(b), the objectness map effectively shows the presence of multiple objects. The bounding boxes filter in Eq. (4) thus tends to select bounding boxes covering all the objects. As a consequence, the selected bounding boxes would be larger than the ground truth bounding box.



(a)

(b)

Fig. 11. Examples of detected object bounding boxes. The three images in each column are the source image, objectness map, and the detected bounding box, respectively. The green bounding box represents the ground truth, and the red bounding box represents the bounding box with highest INT-UION score among the selected bounding boxes $\mathcal{B}_I$. (a) shows the good cases and (b) shows the bad cases.

2) Because of occlusion, some objects are divided into two parts. This makes the generated objectness map focus on the part with higher discriminative ability, *e.g.,* the head of birds. In such case, the selected bounding box would be smaller than the ground truth one.

D. Impact of Parameter K

K is an important parameter in the graph-based object localization in Sec III-D. We evaluate the impact of K on object Detection Rate (DR) and running-time. The related results are summarized in Table IV, where DR is computed using the ranked top-5 bounding boxes.

As shown in Table IV, larger K reasonably corresponds to more running time. It also shows that, larger K does not constantly improve the detection rate, because too large K may take many noisy images into consideration. From Table IV, we observe that K has marginal impact on detection rate, and K=4 is a good trade-off between detection rate and running time. Although Table IV is the analysis based on CUB-200-2011, we observe that various K have slightly changed for detection rate. Therefore, we set $K = 4$ in the following experiments for all datasets.

E. Performance of Deep Features

In this section, we first evaluate different deep features and then show the performance of different feature combination strategies.

TABLE IV
THE IMPACT OF PARAMETER K

K	Running-time (ms)	DR (%)
2	6.7	86.9
4	15.4	87.3
6	28.5	87.3
8	44.9	87

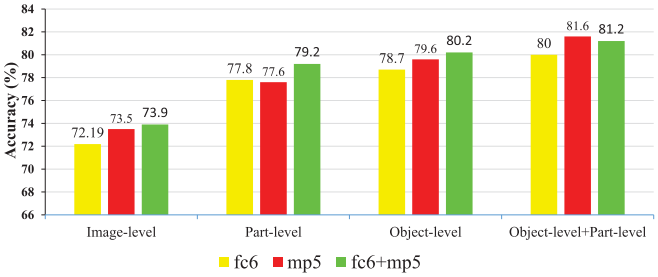


Fig. 12. The performance of different deep features. "mp5" means the feature of *max-pooling of pool5*. "+" denotes feature concatenation.

TABLE V
THE PERFORMANCE OF DIFFERENT METHODS TO GENERATE FINAL FEATURE

Methods	Feature Dimension	Acc.(%)
Max-pooling	512	80.6
Average-pooling	512	81
Concatenation	1024	81.6

The output of each layer in CNN could be treated as an image feature. In [2] and [6], the authors both employ the $fc6$ layer as the feature extractor. In our work, we apply the max-pooling of *pool5* f^{mp5} as the deep feature. We compare the performance of those two features and show the results in Fig. 12. It can be observed that, f^{mp5} outperforms the $fc6$ in most of cases. The concatenation of f^{mp5} and $fc6$ achieves the highest accuracies for image, part, and object level descriptions, respectively. However, when combining the object-level and part-level descriptions, *i.e.*, using our Bi-level description, the f^{mp5} achieves the highest accuracy. Moreover, the f^{mp5} has a lower dimension, *i.e.*, 512D vs. 4096D of $fc6$ feature. Therefore, we use f^{mp5} as the deep feature in the following experiments.

With the object-level and part-level features, we can generate the fused feature with three methods, *i.e., max-pooling*, *average-pooling*, and *concatenation*. We hence evaluate the performance of each method, and choose the best one. As shown in Table V, the *concatenation* achieves the best accuracy. The reason might be because different levels of descriptions complementarily describe the objects from different views, *e.g.*, the global object and the distinctive parts. Therefore, concatenation could better preserve their cues.

F. Performance of Fine-Grained Visual Categorization

In this section, we make comparison with the existing methods on three public datasets.

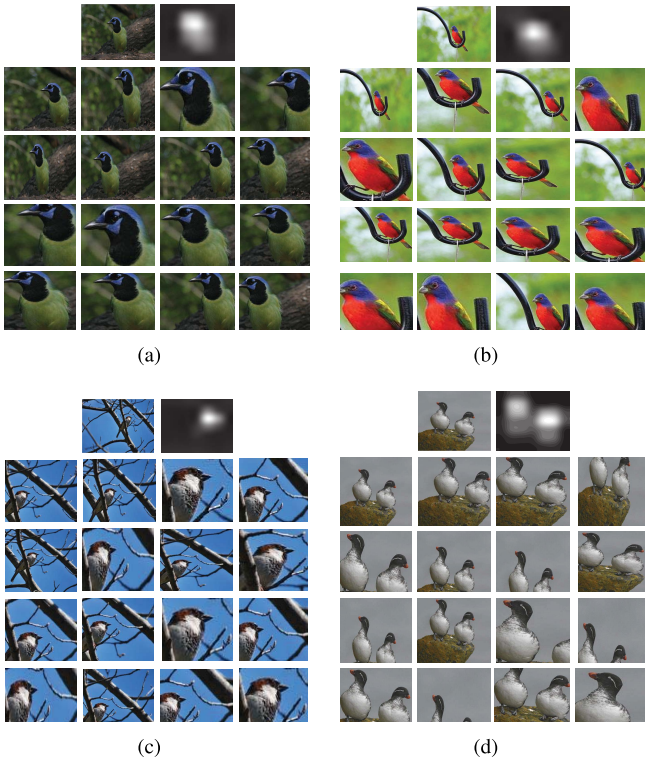


Fig. 13. The original image, objectness map, and selected bounding boxes with $\theta = [0.1, 0.2, 0.3, 0.4, 0.5]$. To make the illustration concise, we present the top 16 bounding boxes. (a) and (b) show two good cases. (c) and (d) show two bad cases.

1) CUB-200-2011: The experimental results and comparisons on CUB-200-2011 are summarized in Table VI.[2] From Table VI, we could see that both the object-level and part-level descriptions outperform the baseline image-level description. The improvements demonstrate that the proposed descriptions are more discriminative than the description from global image. By combining the two descriptions, the AutoBD achieves the classification accuracy of 81.6%, which is 2% and 3.4% better than the object-level and part-level representations, respectively.

Many researchers have reported results on CUB-200-2011 [12]. Existing methods not using bounding box annotations could be classified into two classes: image-level description and part-based description. For example, Lin *et al.* [15], Jaderberg *et al.* [42], and Ge *et al.* [14] describe the fine-grained object only from the whole image and ignore the description from local part. Xiao *et al.* [5] and Simon and Rodner [4] first infer the local parts and then construct a part-based model to describe the objects. Therefore, our method is more similar to the studies by Xiao *et al.* [5] and Simon *et al.* [4]. Only using the image-level labels, the AutoBD achieves the classification accuracy of 81.6%, which outperforms studies of Xiao *et al.* [5] and Simon and Rodner [4] by 4.3% and 0.6%, respectively. Note that, AutoBD also obtains similar performance with [43], which use extra bounding box annotations. The experimental results on CUB-200-2011 clearly demonstrate the high discriminative power of AutoBD.

[2]Methods [4], [5], [14], [15], [42] are categorized into two groups based on their motivation. Methods of [4], [5], and [14] are motivated to generate a powerful description with existing deep networks, e.g., VGG-19. Methods of [15] and [42] target to propose a powerful end-to-end network to describe the fine-grained objects.

TABLE VI

THE CLASSIFICATION ACCURACY ON CUB-200-2011. "BBOX" REFERS TO USING THE GROUND TRUTH BOUNDING BOX. "–" REFERS TO NOT USE ANY ANNOTATIONS[3]

Methods	Training	Testing	Acc.(%)
Krause *et al.* [43]	Bbox	-	82
Xiao *et al.* [5]	-	-	77.9
Ge *et al.* [14]	-	-	77.3
Simon *et al.* [4]	-	-	81.01
Jaderberg *et al.*[42]	-	-	84.1
Lin *et al.* [15]	-	-	84.1
Image-level	-	-	73.5
Object-level	-	-	79.6
Part-level	-	-	78.2
AutoBD	-	-	**81.6**

[3]In the discussion section, we have made some discussion for the difference and relationship between our AutoBD and methods [42], [15].

2) Car196: The results on Car196 [16] are shown in Table VII.[4] Among Table VII, Krause *et al.* [43] is proposed with the bounding box annotations, and Gosselin *et al.* and Gosselin *et al.* [44] are both proposed only with image-level labels. As Table VII shows, using the part-level description achieves a classification accuracy of 84%, which is lower than the baseline image-level description. The reason might be because: 1) on the images of Car-196, the cars commonly occupy the majority part of the image, and 2) different cars have similar parts like tires, windows, *etc.* Thus, parts are not discriminative enough for cars. However, the object-level description still achieves 2.4% higher accuracy than the baseline. It is also interesting to find that, by combining the object-level and part-level descriptions, the AutoBD achieves the classification accuracy of 88.9%, which outperforms most of the state-of-the-art methods.

3) Birdsnap: As the AutoBD only need image-level labels for training, we expect that it could be easily applied to large-scale datasets. Therefore, we also evaluate the AutoBD on the Birdsnap dataset [17], which is a relatively large-scale dataset for fine-grained visual categorization. Experimental results are summarized in Table VIII. In [17], the authors describe the objects by POOF [1]. With the manually labeled parts for training and testing, Berg *et al.* [17] achieve the classification accuracy of 79.9%. When only using the labeled parts for training, Berg *et al.* [17] achieve the classification accuracy of 48.8%.

The results in Table VIII show that using the model pre-trained on ImageNet achieves the classification accuracy of 42.9%. It is only 6% lower than the 48.8% in [17], which is achieved by using the bounding boxes and parts annotations. After fine-tuning the pre-trained model on Birdsnap dataset, the classification accuracy raises to 58.5%. As the Table VIII

[4]Methods [15], [43], [44] are categorized into two groups based on whether they use bounding annotations.

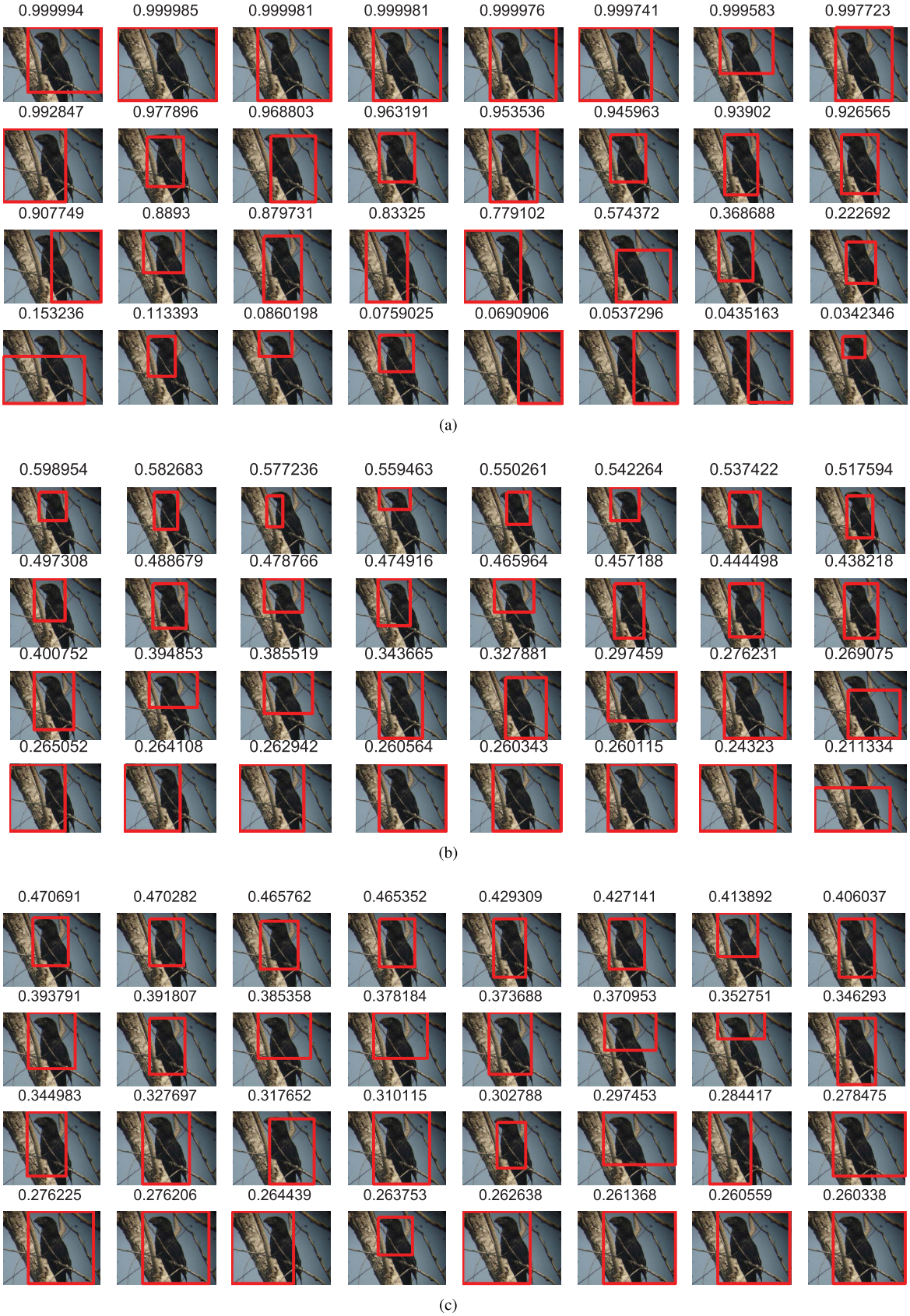


Fig. 14. The bounding boxes ranked with scores computed with different criteria. The red rectangle represents a bounding box and the floating number represents the corresponding score of $probScore(\cdot)$ in (a), $mapScore(\cdot)$ in (b), or $objScore(\cdot)$ in (c), respectively. $objScore(\cdot)$ selects the most accurate bounding boxes.

TABLE VII

COMPARISON OF THE CLASSIFICATION ACCURACY ON CAR-196[5]

Methods	Training	Testing	Acc.(%)
Krause [43]	Bbox	-	92.6
Gosselin *et al.* [44]	-	-	82.7
Lin *et al.* [15]	-	-	91.3
Image-level	-	-	84.5
Object-level	-	-	86.9
Part-level	-	-	84
AutoBD	-	-	**88.9**

[5]In the discussion section, we have made some discussion for the difference and relationship between our AutoBD and method [15].

TABLE VIII

COMPARISON OF THE CLASSIFICATION ACCURACY ON BIRDSNAP. "BBOX+PARTS" REFERS TO USING THE GROUND TRUTH PART ANNOTATIONS AND BOUNDING BOX. "-PRETRAINED" REFERS TO USE THE CNN WHICH WAS TRAINED ON IMAGENET TO EXTRACT THE DEEP FEATURE

Methods	Training	Testing	Acc.(%)
Berg *et al.* [17]	Bbox+Parts	Bbox+Parts	79.9
Berg *et al.* [17]	Bbox+Parts	-	48.8
Berg *et al.* [17]	Bbox+Parts	-	66.6[6]
Image-level pretrained	-	-	42.9
Image-level finetuned	-	-	58.5
Object-level	-	-	66.3
Part-level	-	-	61.5
AutoBD	-	-	**68**

[6]This classification accuracy is achieved by further adding the spatio-tempporal prior

shows, the part-level description further improves the classification accuracy to 61.5%. The object-level description obtains an even more significant improvement. Combining the two representations, AutoBD achieves the classification accuracy of 68%. This is significantly better than the 48.8% and 66.62% in [17], as well as the fine-tuned CNN. This experiments hence clearly show the discriminative power and scalability of AutoBD.

G. Discussions

From Table VI and Table VII, we could see that the Bilinear CNN (B-CNN) [15] and spatial transformer networks (STN) [42] both achieve higher classification accuracies than our method. It should be noted that, B-CNN and STN use better baseline CNN than our VGG-19 network, *e.g.,* performance: 84% [15] and 82.3% [42] vs. 73.5% of VGG19.

It should be noted that, better baseline model could be used to further boost the performance of our approach. However, STN is difficult to train and converge based with the weak global image annotations. The dimensionality of descriptors generated by B-CNN is quite high, *e.g.,* 250,000, making it also not applicable to our approach. To verify our method could obtain higher performance with more powerful baseline model, we tested our approach with GoogleNet with BatchNormalization and summarized the results in Table IX. As shown in Table IX, when using GoogleNet with Batch-Normalization [45] as the baseline model, AutoBD obtains substantially higher performance, *i.e.,* improved from 81.6% to 83.1%. Our future work will study more efficient and compact descriptors to further improve the performance of AutoBD.

TABLE IX

CLASSIFICATION ACCURACY (%) ON CUB-200-2011 WHEN USING VGG19 AND GOOGLENET WITH BATCHNORMALIZATION(BN) AS THE BASELINE MODEL, RESPECTIVELY

Methods	VGG19	GoogleNet+BN [45]
Image-level	73.5	**76.9**
Object-level	79.6	**82.0**
Part-level	78.2	**79.7**
AutoBD	81.6	**83.1**

In this paper, we assume that the image only contains one object. This assumption is reasonable because the groundtruth bounding boxes only cover one object. However, as discussed in Sec. IV-C, bounding box filtering does not work well for images containing multiple objects. This issue could be effectively addressed if we know how many objects present in the image. From Fig. 11(b), we could observe that the objectness map effectively shows the presence of multiple objects, *e.g.,* present several disjoint regions with high response. Therefore, given an image along with its objectness map, we could obtain estimate the number of object based on the objectness map. Then, different bounding boxes could be selected for each of the salient regions.

V. CONCLUSIONS AND FUTURE WORK

This work is motivated to improve the scalability and conquer the dependencies on annotations of object parts or bounding boxes in fine-grained visual classification. To achieve this, we propose a robust and discriminative visual description named Automated Bi-level Description (AutoBD). "Bi-level" denotes two complementary part-level and object-level visual descriptions, respectively. The part-level description is extracted from a region saliently representing the visual distinctiveness, and the object-level description is extracted from the object bounding boxes. These two representations can be generated only with image-level labels, hence we call AutoBD as "automated". The easily acquired image-level labels make AutoDB suitable for large-scale fine-grained visual categorization tasks. Our experiments show that, although only using the image-level labels, AutoBD outperforms the recent works on CUB-200-2011 and Car-196. AutoBD also achieves promising performance on the large-scale BirdSnap dataset.

In our future work, we plan to further improve AutoBD. Our current implementation uses bounding boxes generated by Selective Search as the basis for region selection. However, the bounding box generation is time-consuming and most of the generated bounding boxes are useless. Therefore, generating more accurate candidates regions directly from objectness map or the CNN output is potential to significantly accelerate AutoBD. Additionally, the object-level and part-level descriptions are closely related with each other, *e.g.*, the part-regions should always be inside the object bounding box. However, these two descriptions are learned separately in current AutoBD. Investigating how to take advantage of

their relationship might further improve the performance of AutoBD.

REFERENCES

[1] T. Berg and P. N. Belhumeur, "POOF: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 955–962.

[2] S. Branson, G. Van Horn, S. Belongie, and P. Perona, "Bird species categorization using pose normalized deep convolutional nets," in *Proc. BMVC*, Jun. 2014. [Online]. Available: http://arxiv.org/abs/1406.2952

[3] D. Lin, X. Shen, C. Lu, and J. Jia, "Deep LAC: Deep localization, alignment and classification for fine-grained recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1666–1674.

[4] M. Simon and E. Rodner, "Neural activation constellations: Unsupervised part model discovery with convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1143–1151.

[5] T. Xiao, Y. Xu, K. Yang, J. Zhang, Y. Peng, and Z. Zhang, "The application of two-level attention models in deep convolutional neural network for fine-grained image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 842–850.

[6] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *Computer Vision—ECCV*. Cham, Switzerland: Springer, 2014, pp. 834–849.

[7] N. Zhang, R. Farrell, F. Iandola, and T. Darrell, "Deformable part descriptors for fine-grained recognition and attribute prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 729–736.

[8] N. Zhang, E. Shelhamer, Y. Gao, and T. Darrell. (Nov. 2015). "Fine-grained pose prediction, normalization, and recognition." [Online]. Available: https://arxiv.org/abs/1511.07063

[9] J. Liu and P. N. Belhumeur, "Bird part localization using exemplar-based models with enforced pose and subcategory consistency," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2520–2527.

[10] H. Yao, D. Zhang, J. Li, J. Zhou, S. Zhang, and Y. Zhang, "DSP: Discriminative spatial part modeling for fine-grained visual categorization," *Image Vis. Comput.*, vol. 63, pp. 24–37, Jul. 2017.

[11] H. Yao, S. Zhang, Y. Zhang, J. Li, and Q. Tian, "Coarse-to-fine description for fine-grained visual categorization," *IEEE Trans. Image Process.*, vol. 25, no. 10, pp. 4858–4872, Oct. 2016.

[12] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," Comput. Neural Syst., Tech. Rep. CNS TR-2011-001, 2011.

[13] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Apr. 2013.

[14] Z. Ge, C. McCool, C. Sanderson, and P. Corke, "Subset feature learning for fine-grained category classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2015, pp. 46–52.

[15] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN models for fine-grained visual recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1449–1457.

[16] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Jun. 2013, pp. 554–561.

[17] T. Berg, J. Liu, S. W. Lee, M. L. Alexander, D. W. Jacobs, and P. N. Belhumeur, "Birdsnap: Large-scale fine-grained visual categorization of birds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 2019–2026.

[18] E. Gavves, B. Fernando, C. G. M. Snoek, A. W. M. Smeulders, and T. Tuytelaars, "Fine-grained categorization by alignments," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1713–1720.

[19] L. Xie, Q. Tian, R. Hong, S. Yan, and B. Zhang, "Hierarchical part matching for fine-grained visual categorization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 1641–1648.

[20] Y. Chai, E. Rahtu, V. Lempitsky, L. Van Gool, and A. Zisserman, "TriCoS: A tri-level class-discriminative co-segmentation method for image classification," in *Proc. Comput. Vis.-ECCV*, 2012, pp. 794–807.

[21] Y. Chai, V. Lempitsky, and A. Zisserman, "Symbiotic segmentation and part localization for fine-grained categorization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2013, pp. 321–328.

[22] Y. Souri and S. Kasaei, "Fast bird part localization for fine-grained categorization," in *Proc. 3rd Workshop Fine-Grained Vis. Categorization (FGVC3) CVPR* , Jun. 2015.

[23] C. Göering, E. Rodner, A. Freytag, and J. Denzler, "Nonparametric part transfer for fine-grained recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 2489–2496.

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2014, pp. 580–587.

[25] A. Criminisi, J. Shotton, and E. Konukoglu, "Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning," *Found. Trends Comput. Graph. Vis.*, vol. 7, nos. 2–3, pp. 81–227, Feb. 2012.

[26] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440.

[27] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2008, pp. 1–8.

[28] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, Jun. 2005, pp. 886–893.

[29] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Proc. Workshop Stat. Learn. Comput. Vis. (ECCV)*, vol. 1. 2004, pp. 1–16.

[30] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proc. Comput. Vis.–ECCV*, 2010, pp. 143–156.

[31] L. Bo, X. Ren, and D. Fox, "Kernel descriptors for visual recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 244–252.

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

[33] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.

[34] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 675–678.

[35] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008.

[36] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. ICLR*, 2015, pp. 1–14.

[37] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Sep. 2009.

[38] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, "Novel dataset for fine-grained image categorization," in *Proc. CVPR Workshop Fine-Grained Vis. Categorization (FGVC)*, vol. 2. 2011, p. 1.

[39] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. (Jun. 2013). "Fine-grained visual classification of aircraft." [Online]. Available: https://arxiv.org/abs/1306.5151

[40] R. G. Cinbis, J. Verbeek, and C. Schmid, "Segmentation driven object detection with fisher vectors," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2968–2975.

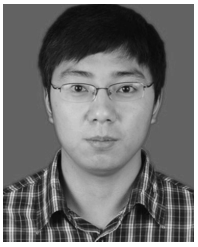
[41] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2189–2202, Nov. 2012.

[42] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2008–2016.

[43] J. Krause, H. Jin, J. Yang, and L. Fei-Fei, "Fine-grained recognition without part annotations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5546–5555.

[44] P.-H. Gosselin, N. Murray, H. Jégou, and F. Perronnin, "Revisiting the fisher vector for fine-grained classification," *Pattern Recognit. Lett.*, vol. 49, pp. 92–98, Aug. 2014.

[45] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.



Hantao Yao received the B.S. degree from Xidian University, Xi'an, China, in 2012. He is currently pursuing the Ph.D. degree with the Institute of Computing Technology, University of Chinese Academy of Sciences, Beijing, China. His current research interests are compute vision and image retrieval.

Shiliang Zhang (M'15) received the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, in 2012. He was a Post-Doctoral Scientist with NEC Labs America and a Post-Doctoral Research Fellow with The University of Texas at San Antonio. He is currently a tenure-track Assistant Professor with the School of Electronic Engineering and Computer Science, Peking University.

His research interests include large-scale image retrieval and computer vision for autonomous driving. His research was supported by the National 1000 Youth Talents Plan and Natural Science Foundation of China. He received the National 1000 Youth Talents Plan of China, Outstanding Doctoral Dissertation Awards from both Chinese Academy of Sciences and Chinese Computer Federation, the President Scholarship by Chinese Academy of Sciences, the NEC Laboratories America Spot Recognition Award, and the Microsoft Research Fellowship. He was a recipient of the Top 10% Paper Award in the IEEE MMSP 2011.

Chenggang Yan received the B.S. degree in computer science from Shandong University in 2008 and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, in 2013. He was an Assistant Research Fellow with Tsinghua University. He is currently a Professor with Hangzhou Dianzi University. He has authored or co-authored over 30 refereed journal and conference papers. His research interests include intelligent information processing, machine learning, image processing, computational biology, and computational photography. As a co-author, he received the Best Paper Awards in International Conference on Game Theory for Networks 2014 and SPIE/COS Photonics Asia Conference 9273 2014 and the Best Paper Candidate in the International Conference on Multimedia and Expo 2011.

Yongdong Zhang (M'08–SM'13) received the Ph.D. degree in electronics engineering from Tianjin University, Tianjin, China, in 2002. He is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. He has authored over 100 refereed journal and conference papers. His current research interests are in the fields of multimedia content analysis and understanding, multimedia content security, video encoding, and streaming media technology. He was a recipient of the best paper awards in PCM 2013, ICIMCS 2013, and ICME 2010, and the Best Paper Candidate in ICME 2011. He serves as an Editorial Board Member of the *Multimedia Systems* Journal and the IEEE TRANSACTIONS ON MULTIMEDIA.

Jintao Li received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 1989. He is currently a Professor with the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include multimedia technology, virtual reality technology, and pervasive computing.

Qi Tian (F'15) was a Visiting Professor with NEC Laboratories of America in 2003 and a Visiting Scholar with the MIAS Center, University of Illinois at Urbana–Champaign (UIUC), in 2007. He was a tenure-track Assistant Professor from 2002 to 2008 and a tenured Associate Professor from 2008 to 2012. From 2008 to 2009, he took one-year Faculty Leave with the Media Computing Group, Microsoft Research Asia, as a Lead Researcher. He is currently a Full Professor with the Department of Computer Science, The University of Texas at San Antonio (UTSA).

He received the B.E. degree in electronic engineering from Tsinghua University in 1992, the M.S. degree in ECE from Drexel University in 1996 and the Ph.D. degree in ECE from UIUC in 2002. He has published over 310 refereed journal and conference papers. His research interests include multimedia information retrieval, computer vision, pattern recognition, and bioinformatics. He was the co-author of the Best Paper in ACM ICMR 2015, the Best Paper in PCM 2013, the Best Paper in MMM 2013, the Best Paper in ACM ICIMCS 2012, the Top 10% Paper Award in MMSP 2011, and the Best Student Paper in ICASSP 2006, and co-author of the Best Student Paper Candidate in ICME 2015 and the Best Paper Candidate in PCM 2007.

Dr. Tian has served as a Founding Member of the International Steering Committee for the ACM International Conference on Multimedia Retrieval from 2009 to 2014. He has been an ACM Multimedia Conference Review Committee Member since 2009. He has served as an International Steering Committee Member for ACM MIR from 2006 to 2010 and a Best Paper Committee Member for ACM Multimedia 2009, ACM ICIMCS 2013, ICME 2006 and 2009, PCM 2012, and the IEEE International Symposium on Multimedia 2011. He will/has served as the General Chair for ACM Multimedia 2015, a Program Coordinator for ACM Multimedia 2009, and the Program Chair for various international conferences, including ACM CIVR 2010, ACM ICMCS 2009, MMM 2010, IMAI 2007, VIP 2007 and 2008, and MIR 2005. He has also served in various organization committees as the Panel and Tutorial Chair, the Publicity Chair, the Special Session Chair, and the Track Chair in numerous ACM and IEEE conferences, such as ACM Multimedia, VCIP, PCM, CIVR, and ICME, and served as a TPC Member for prestigious conferences, such as ACM Multimedia, SIGIR, ICCV, and KDD.

His research projects are funded by ARO, NSF, DHS, Google, FXPAL, NEC, SALSI, CIAS, Akiira Media Systems, HP, and UTSA. He has been a member of ACM since 2004. He received 2014 Research Achievement Awards from the College of Science, UTSA. He received the 2010 ACM Service Award. He is an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the *Multimedia System* Journal, and in the Editorial Board of the *Journal of Multimedia* and the *Journal of Machine Vision and Applications*. He is the Guest Editor of the IEEE TRANSACTIONS ON MULTIMEDIA and the *Journal of Computer Vision and Image Understanding*.