Learning Attentive Pairwise Interaction for Fine-Grained Classification

Peiqin Zhuang,∗1,2 Yali Wang,∗1,2 Yu Qiao†1,2

1ShenZhen Key Lab of Computer Vision and Pattern Recognition, SIAT-SenseTime Joint Lab, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
2SIAT Branch, Shenzhen Institute of Artificial Intelligence and Robotics for Society

Abstract
Fine-grained classification is a challenging problem, due to subtle differences among highly-confused categories. Most approaches address this difficulty by learning discriminative representation of individual input image. On the other hand, humans can effectively identify contrastive clues by comparing image pairs. Inspired by this fact, this paper proposes a simple but effective Attentive Pairwise Interaction Network (API-Net), which can progressively recognize a pair of fine-grained images by interaction. Specifically, API-Net first learns a mutual feature vector to capture semantic differences in the input pair. It then compares this mutual vector with individual vectors to generate gates for each input image. These distinct gate vectors inherit mutual context on semantic differences, which allow API-Net to attentively capture contrastive clues by pairwise interaction between two images. Additionally, we train API-Net in an end-to-end manner with a score ranking regularization, which can further generalize API-Net by taking feature priorities into account. We conduct extensive experiments on five popular benchmarks in fine-grained classification. API-Net outperforms the recent SOTA methods, i.e., CUB-200-2011 (90.0%), Aircraft (93.9%), Stanford Cars (95.3%), Stanford Dogs (90.3%), and NABirds (88.1%).

1 Introduction
Over the past years, CNNs have achieved remarkable successes for visual recognition (He et al. 2016; Huang et al. 2017). However, these classical models are often limited to distinguish fine-grained categories, due to highly-confused appearances. Subsequently, a number of fine-grained frameworks have been proposed by finding key part regions (Fu, Zheng, and Mei 2017; Zheng et al. 2017; Yang et al. 2018), learning patch relations (Lin, RoyChowdhury, and Maji 2015; Cai, Zuo, and Zhang 2017; Yu et al. 2018), etc. But most of them take individual image as input, which may limit their ability to identify contrastive clues from different images for fine-grained classification.

∗ Equally-contributed first authors
† Corresponding author
Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Figure 1: Motivation (Best view in color). Caspian Tern and Elegant Tern are two highly-confused bird species. Humans often distinguish them by pairwise comparison, instead of checking each individual image alone. First, humans would exploit contrastive clues (e.g. body and mouth) from the image pair, and then check each image with these mutual contexts to further discover distinct attentions on each image. Finally, humans recognize both images via comparing subtle differences jointly. To mimic this capacity, we propose a novel API-Net. More details can be found in Section

On the contrary, humans often recognize fine-grained objects by comparing image pairs (Bruner 2017), instead of checking single images alone. For example, Caspian Tern and Elegant Tern are two highly-confused bird species, as shown in Fig. 1. If we only check individual image alone, it is difficult to recognize which categories it belongs to, especially when the bird is self-occluded with a noisy back-
ground. Alternatively, we often take a pair of images together, and summarize contrastive visual appearances as context, e.g., body, mouth and etc. Then, we check individual images with these mutual contexts, so that we can further understand distinct aspects of each image, e.g.,

the body of the bird is an important part for the left image, while the mouth of the bird is a key characteristic for the right image. With such discriminative guidance, we pay different attentions to the body and the mouth of two birds. Note that, for each bird in the pair, we not only check its prominent part but also have a glance at the distinct part found from the other bird. This comparative interaction can effectively tell us that, \textit{Caspian Tern} has a fatter body, while \textit{Elegant Tern} has a more curved mouth. Consequently, we recognize both images jointly.

To mimic this capacity of human beings, we introduce a novel Attentive Pairwise Interaction Network (API-Net) for fine-grained classification. It can adaptively discover contrastive clues from a pair of fine-grained images, and attentively distinguish them via pair interaction.

More specifically, API can effectively recognize two fine-grained images, by a progressive comparison procedure like human beings. To achieve this goal, API-Net consists of three submodules, i.e., mutual vector learning, gate vector generation, and pairwise interaction. By taking a pair of fine-grained images as input,

API-Net first learns a mutual vector to summarize contrastive clues of input pair as context. Then, it compares mutual vector with individual vectors. This allows API-Net to generate distinct gates, which can effectively highlight semantic differences respectively from the view of each individual image.

Consequently, API-Net uses these gates as discriminative attentions to perform pairwise interaction. In this case, each image can generate two enhanced feature vectors, which are activated respectively from its own gate vector and the gate vector of the other image in the pair. Via an end-to-end training manner with a score-ranking regularization, API-Net can promote the discriminative ability of all these features jointly with different priorities. Additionally, it is worth mentioning that, one can easily embed API into any CNN backbone for fine-grained classification, and flexibly unload it for single-input test images without loss of generalization capacity. Such plug-and-play property makes API as a preferable choice in practice. Finally, we evaluate API-Net on five popular benchmarks in fine-grained recognition, namely CUB-200-2011, Aircraft, Stanford Cars, Stanford Dogs and NABirds. The extensive results show that, API-Net achieves the state-of-the-art performance on all these datasets.

2 Related Works

A number of research works have been recently proposed for fine-grained classification. In the following, we mainly summarize and discuss those related works.

Object Parts Localization. These approaches mainly utilize the pre-defined bounding boxes or part annotations to capture visual details in the local regions (Zhang et al. 2013). However, collecting such annotations is often labor-intensive or infeasible in practice. Hence, several weakly-supervised localization approaches have been recently proposed by designing complex spatial attention mechanisms (e.g., RA-CNN (Fu, Zheng, and Mei 2017), MA-CNN (Zheng et al. 2017)), learning a bank of discriminative filters (Wang, Morariu, and Davis 2018), guiding region detection with multi-agent cooperation (Yang et al. 2018), etc. But these approaches mainly focus on mining local characteristics of fine-grained images. Consequently, they may lack the capacity of discriminative feature learning.

Discriminative Feature Learning. To learn the representative features, many approaches have been exploited by patch interactions (Lin, RoyChowdhury, and Maji 2015; Cai, Zuo, and Zhang 2017; Yu et al. 2018). A well-known approach is B-CNN (Lin, RoyChowdhury, and Maji 2015), which performs bilinear pooling on the representations of two local patches in an image. Following this direction, several high-order approaches have been proposed via polynomial kernel formulation (Cai, Zuo, and Zhang 2017), cross-layer bilinear representation (Yu et al. 2018), etc. However, these approaches take a single image alone as input, while neglecting comparisons between different images, i.e., an important clue to distinguish highly-confused objects.

Metric Learning. Metric learning refers to the method that uses similarity measurements to model relations between image pairs (Kulis and others 2013). It has been widely used in Face Verification (Schroff, Kalenichenko, and Philbin 2015), Person ReID (Hermans, Beyer, and Leibe 2017) and so on. Recently, it has been introduced for fine-grained classification, e.g., triplet loss design (Zhang et al. 2016b), pairwise confusion regularization (Dubey et al. 2018a), multi-attention multi-class constraint (Sun et al. 2018), etc. However, these approaches mainly leverage metric learning to improve sample distributions in the feature space. Hence, they often lack the adaptation capacity, w.r.t., how to discover visual differences between a pair of images.

Different from previous approaches, our API-Net can adaptively summarize contrastive clues, by learning a mutual vector from an image pair. As a result, we can leverage it as guidance, and attentively distinguish two fine-grained images via pairwise interaction.

3 Attentive Pairwise Interaction

In this section, we describe Attentive Pairwise Interaction Network (API-Net) for fine-grained classification. Our design is partially inspired by the observation that, instead of learning visual concepts alone with individual image, humans often compare a pair of images jointly to distinguish subtle differences between similar objects (Bruner 1977).

To imitate this capacity, we simultaneously take a pair of images as input to API-Net, and progressively distinguish them by three elaborative submodules, i.e., mutual vector learning, gate vector generation, and pairwise interaction. The whole framework is demonstrated in Fig. 2.

Mutual Vector Learning. First, we feed two fine-grained images into a CNN backbone, and extract their $D$-dimension
1. Mutual Vector Learning

Figure 2: The framework of API-Net (Best view in color). API can progressively recognize a pair of fine-grained images, based on a novel human-like learning procedure. It consists of three submodules. 1. **Mutual Vector Learning**. API learns a mutual vector \( x_m \) from individual \( x_1 \) and \( x_2 \) (Eq. 1). In this case, it can summarize contrastive clues in the pair. 2. **Gate Vector Generation**. API further compares \( x_m \) with \( x_1 \) and \( x_2 \), and generates two distinct gate vectors \( g_i \) (Eq. 2). These gates allow API to discover distinct clues respectively from the view of each individual image. 3. **Pairwise Interaction**. API performs pairwise interaction with guidance of gate vectors (Eq. 3-6). Via training API-Net with a score-ranking regularization, we can distinguish all these features jointly with the consideration of feature priorities (Eq. 8-10). Additionally, API is a practical plug-and-play module, i.e., one can combine API with CNNs during training, and flexibly unload it for single-input test images.

feature vectors respectively, i.e., \( x_1 \) and \( x_2 \in \mathbb{R}^D \). Then, we learn a mutual vector \( x_m \in \mathbb{R}^D \) from individual \( x_1 \) and \( x_2 \),

\[
x_m = f_m([x_1, x_2]),
\]

where \( f_m(\cdot) \) is a mapping function of \([x_1, x_2]\), e.g., a simple MLP works well in our experiments. Since \( x_m \) is adaptively summarized from both \( x_1 \) and \( x_2 \), it often contains feature channels which indicate high-level contrastive clues (e.g. body and mouth of two birds in Fig. 1) in the pair.

**Gate Vector Generation**. After learning the mutual vector \( x_m \), we propose to compare it with \( x_1 \) and \( x_2 \). The main reason is that, we should further generate distinct clues respectively from the view of each individual image, in order to distinguish this pair afterwards.

In particular, we perform channel-wise product between \( x_m \) and \( x_i \), so that we can leverage \( x_m \) as guidance to find which channels of individual \( x_i \) may contain contrastive clues. Then, we add a sigmoid function to generate the gate vector \( g_i \in \mathbb{R}^D \).

\[
g_i = \text{sigmoid}(x_m \odot x_i), \quad i \in \{1, 2\}.
\]

As a result, \( g_i \) becomes a discriminative attention which highlights semantic differences with a distinct view of each individual \( x_i \), e.g., the body of the bird is important in one image of Fig. 1 while the mouth of the bird is a key in the other image. In our experiments, we evaluate different gating strategies and show effectiveness of our design.

**Pairwise Interaction**. Next, we perform pairwise interaction by gate vectors. Our design is partially motivated by the fact that, to capture subtle differences in a pair of fine-grained images, human check each image not only with its prominent parts but also with distinct parts from the other image. For this reason, we introduce an interaction mechanism via residual attention,

\[
x_1^{self} = x_1 + x_1 \odot g_1,
\]

\[
x_2^{self} = x_2 + x_2 \odot g_2,
\]

\[
x_1^{other} = x_1 + x_1 \odot g_2,
\]

\[
x_2^{other} = x_2 + x_2 \odot g_1.
\]

As we can see, each individual feature \( x_i \) in the pair produces two attentive feature vectors, i.e., \( x_i^{self} \in \mathbb{R}^D \) is highlighted by its own gate vector, and \( x_i^{other} \in \mathbb{R}^D \) is activated by the gate vector of the other image in the pair. In this case, we enhance \( x_i \) with discriminative clues that come from both images. Via distinguishing all these features jointly, we can reduce confusion in this fine-grained pair.

**Training**. After obtaining four attentive features \( x_i \) in the pair (where \( i \in \{1, 2\} \), \( j \in \{self, other\} \)), we feed them into a softmax classifier,

\[
p_i^j = \text{softmax}(Wx_i^j + b),
\]

where \( p_i^j \in \mathbb{R}^C \) is the prediction score vector, \( C \) is the number of object categories, and \( \{W, b\} \) is the parameter set of classifier. To train the whole API-Net effectively, we design the following loss \( L \) for a pair,

\[
L = L_{ce} + \lambda L_{rk},
\]

where \( L_{ce} \) is a cross entropy loss, and \( L_{rk} \) is a score ranking regularization with coefficient \( \lambda \).

1) **Cross Entropy Loss**. The main loss is the cross entropy loss \( L_{ce} \),

\[
L_{ce} = - \sum_{i \in \{1, 2\}} \sum_{j \in \{self, other\}} y_i^j \log(p_i^j),
\]
where $y_i$ is the one-hot label vector for image $i$ in the pair, $\top$ denotes the vector transposition. By using this loss, API-Net can gradually recognize all the attentive features $x_i^*$, with supervision of label $y_i$.

2) Score Ranking Regularization. Additionally, we introduce a hinge loss as the score ranking regularization $\mathcal{L}_{rk}$.

$$\mathcal{L}_{rk} = \sum_{i \in \{1, 2\}} \max(0, \mathbf{p}_i^{other}(c_i) - \mathbf{p}_i^{self}(c_i) + \epsilon),$$

where $\mathbf{p}_i^{self}(c_i) \in \mathbb{R}$ is the score obtained from the prediction vector $\mathbf{p}_i^{self}$, and $c_i$ is the corresponding index associated with the ground truth label of image $i$. Our motivation of this design is that, $x_i^{\text{self}}$ is activated by its own gate vector. Hence, it should be more discriminative to recognize the corresponding image, compared with $x_i^{\text{other}}$.

To take this knowledge into account, we use $\mathcal{L}_{rk}$ to promote the priority of $x_i^{\text{self}}$, i.e., the score difference $\mathbf{p}_i^{\text{self}}(c_i) - \mathbf{p}_i^{\text{other}}(c_i)$ should be larger than a margin $\epsilon$. With such a regularization, API-Net learns to recognize each image in the pair by adaptively taking feature priorities into account.

3) Pair Construction. Eq. (8) defines the loss for a training pair. Next, we explain how to construct multiple pairs in a batch for end-to-end training. Specifically, we randomly sample $N_{ij}$ classes in a batch. For each class, we randomly sample $N_{im}$ training images. Then, we feed all these images into CNN backbone to generate their feature vectors. For each image, we compare its feature with others in the batch, according to Euclidean distance. As a result, we can construct two pairs for each image, i.e., the intra / inter pair consists of its feature and its most similar feature from intra / inter classes in the batch. This design allows our API-Net for learning to distinguish which are highly-confused or truly-similar pairs. We also investigate different construction strategies in our experiments. Consequently, there are $2 \times N_{ij} \times N_{im}$ pairs in each batch. We pass them into our API module, and summarize the loss $\mathcal{L}$ over all these pairs for end-to-end training.

Testing. We would like to emphasize that, API is a practical plug-and-play module for fine-grained classification. In the training phase, this module can summarize contrastive clues from a pair, which can gradually generalize the discriminative power of CNN representations for each individual image. Hence, in the testing phase, one can simply unload API for single-input test images, without much loss of generalization. Specifically, we feed a test image into the CNN backbone to extract its feature vector $x_i \in \mathbb{R}^{D}$. Then, we directly put $x_i$ into softmax classifier (Eq. 7). The resulting score vector $\mathbf{p}_i \in \mathbb{R}^{C}$ is used for label prediction. By doing so, our testing manner is just the same as that of a plain CNN, which largely boosts the value of API-Net for fine-grained classification.

4 Experiments

Data Sets. We evaluate API-Net on five popular fine-grained benchmarks, i.e., CUB-200-2011 [Wah et al. 2011], Aircraft [Maji et al. 2013], Stanford Cars [Krause et al. 2013], Stanford Dogs [Khosla et al. 2011] and NABirds [Van Horn et al. 2015]. Specifically, CUB-200-2011 / Aircraft / Stanford Cars / Stanford Dogs / NABirds consists of 11,788 / 10,000 / 16,185 / 20,580 / 48,562 images, from 200 bird / 100 airplane / 196 car / 120 dog / 555 bird classes. We use the official train & test splits for evaluation.

Implementation Details. Unless stated otherwise, we implement API-Net as follows. First, we resize each image as $512 \times 512$, and crop a $448 \times 448$ patch as input to API-Net (train: random cropping; test: center cropping). Furthermore, we use ResNet-101 (pretrained on ImageNet) as CNN backbone, and extract the feature vector $x_i \in \mathbb{R}^{2048}$ after global pooling average operation. Second, for all the datasets, we randomly sample 30 categories in each batch. For each category, we randomly sample 4 images. For each image, we find its most similar image from its own class and the rest classes, according to Euclidean distance between features. As a result, we obtain an intra pair and an inter pair for each image in the batch. For each pair, we concatenate $x_1$ and $x_2$ as input to a two-layer MLP, i.e., FC(4096→512)-FC(512→2048). Consequently, this operation generates the mutual vector $x_m \in \mathbb{R}^{2048}$. Finally, we implement our network by PyTorch. For all the datasets, the coefficient $\lambda$ in Eq. (8) is 1.0, and the margin $\epsilon$ in the score-ranking regularization is 0.05. We use the standard SGD with momentum of 0.9, weight decay of 0.0005. Furthermore, the initial learning rate is 0.01 (0.001 for Stanford Dogs), and adopt cosine annealing strategy to adjust it. The total number of training epochs is 100 (50 for Stanford Dogs). Besides, during training phase, we freeze the conv layers and only train the newly-added fully-connected layers in the first 8 epochs (12 epochs for Standard Dogs).

4.1 Ablation Studies

To investigate the properties of our API-Net, we evaluate its key designs on CUB-200-2011. For fairness, when we explore different strategies of one design, others are set as the basic strategy stated in the implementation details.

Basic Methods. First of all, we compare our API-Net with Baseline, i.e., the standard ResNet-101 without our API design. As expected, API-Net largely outperforms it in Table 1, showing the effectiveness of API module.

Mutual Vector. To demonstrate the essentiality of $x_m$, in Eq. (1), we investigate different operations to generate it. (I) Individual Operation. The key of $x_m$ is to learn mutual information from both images in the pair. For comparison, we introduce a baseline without it. Specifically, we replace mutual learning in Eq. (1) by individual learning $x_i = f_m(x_i)$, and use $x_i$ to generate the gate vector $g_i = \text{sigmoid}(\tilde{x}_i)$ where $i \in \{1, 2\}$. (II) Compact Bilinear Pooling Operation. We generate $x_m$ by compact bilinear pooling (Gao et al. 2016) between $x_1$ and $x_2$. (III) Elementwise Operations. We perform a number of widely-used elementwise operations to generate $x_m$, including SubtractSquare $x_m = (x_1 - x_2)^2$, Sum $x_m = x_1 + x_2$, and Product $x_m = x_1 \odot x_2$. (IV) Weight Attention Operation. We use a two-layer MLP to generate the weight of $x_1$ and $x_2$, i.e., $[w_1, w_2] = \text{softmax}(f_w([x_1, x_2]))$, where the output dimension of the 1st FC layer is 512. Then, we use the normalized weight vector as attention to generate the mutual
Table 1: Comparison with basic methods. Baseline is the standard ResNet-101 without our API design.

| Method          | Baseline | API-Net |
|-----------------|----------|---------|
| Accuracy        | 85.4     | 88.6    |

Table 2: Different operations of mutual vector (CUB-200-2011). More details can be found in Section 4.1

| Mutual Vector | Accuracy |
|---------------|----------|
| Individual    | 87.9     |
| Compact Bilinear Pooling | 88.2     |
| Subtract Square | 88.3    |
| Sum           | 88.5     |
| Product       | 88.4     |
| Weight Attention | 88.4   |
| MLP           | 88.6     |

Table 3: Different strategies of gate vector and interaction (CUB-200-2011). More details can be found in Section 4.1

| Gate Vector | Accuracy | Interaction | Accuracy |
|-------------|----------|-------------|----------|
| Single      | L_{cc}   | L_{cc} + L_{rk} | 88.6     |
| Pair        | 88.6     | 88.1        |

Table 4: Different strategies of pair construction (CUB-200-2011). Random: We randomly sample 240 image pairs in a batch. Class-Image: We sample 240 image pairs in a batch, according to class (i.e., Intra / Inter) and Euclidean distance (i.e., Similar(S) / Dissimilar(D)). This can generate 8 Class-Image settings. For example, we randomly sample 30 classes in a batch. For each class, we randomly sample 4 images. For each image, we construct 2 pairs, i.e., the intra / inter pair consists of this image and its most similar image from intra / inter classes. This is denoted as Intra(S) & Inter(S).

| Pair Construction | Intra | Inter | Accuracy |
|-------------------|-------|-------|----------|
| Random            | -     | -     | 86.4     |
|                   | D     | D     | 85.4     |
|                   | S     | S     | 87.2     |
|                   | D     | S     | 87.1     |
|                   | S     | D     | 87.0     |
|                   | D     | D     | 87.4     |
|                   | S     | S     | 88.3     |
|                   | S     | S     | 88.6     |

Class-Image

| Class Size (N_{cl}) | N_{im}=10 | N_{im}=20 | N_{im}=30 |
|---------------------|-----------|-----------|-----------|
| Accuracy            | 83.5      | 87.0      | 88.6      |
| Image Size (N_{im}) | N_{cl}=2  | N_{cl}=3  | N_{cl}=4  |
| Accuracy            | 88.1      | 88.2      | 88.6      |
|                     | (24, 5)   | (30, 4)   | (40, 3)   |
| Accuracy            | 87.7      | 88.6      | 88.2      |

Table 5: Class Size & Image Size (CUB-200-2011). After choosing Intra(S) & Inter(S) in the Class-Image rule, we further explore the number of sampled classes and images in each batch. When varying the class/image size, we fix the image/class size as 4/30.
Table 6: Comparison with The-State-of-The-Art (CUB-200-2011). Extra S.: Extra Supervision.

| Method                  | Backbone | Extra S. | Accuracy |
|-------------------------|----------|----------|----------|
| DeepLoc [Li et al. 2015] | VGGNet-19 | Yes      | 89.4     |
| QAT-ResNet [Zhu et al. 2014] | VGGNet-19 | Yes      | 89.4     |
| PAF-CNN [Krane et al. 2015] | VGGNet-19 | Yes      | 89.4     |
| MG-CNN [Wang et al. 2015] | VGGNet-19 | Yes      | 89.4     |
| T-CNN [Li et al. 2016a] | VGGNet-16 | No       | 89.4     |
| TA-FCN [Yin et al. 2018a] | ResNet-50 | Yes      | 89.4     |
| HNB [Chen et al. 2018a] | VGGNet-19 | Yes      | 89.4     |
| FSP [Zhang et al. 2015] | VGGNet-19 | No       | 89.4     |
| Grassmann Pool [Wei et al. 2018] | VGGNet-16 | No       | 89.4     |
| KI [Cui et al. 2016] | VGGNet-16 | No       | 89.4     |
| HBP [Wu et al. 2018] | VGGNet-16 | No       | 89.4     |
| G"DNet [Wang, Li, and Zhang 2017] | VGGNet-16 | No       | 89.4     |
| MGCNN [Wang et al. 2018c] | VGGNet-19 | No       | 89.4     |
| B-CNN [Lin, RoyChowdhury, and Maji 2015] | VGGNet-19 | No       | 89.4     |
| RACNN [Fu, Zheng, and Mei 2017] | VGGNet-19 | No       | 89.4     |
| MACNN [Zhang et al. 2017] | VGGNet-19 | No       | 89.4     |
| Deep KSSP [Rajagopal et al. 2018a] | VGGNet-19 | No       | 89.4     |
| STCNN [Liu et al. 2018b] | VGGNet-19 | No       | 89.4     |
| PC [Dabney et al. 2018a] | DenseNet-161 | No | 89.4     |
| Our API-Net | ResNet-101 | No | 89.4     |
| Our API-Net | DenseNet-161 | No | 89.4     |

Table 7: Comparison with The-State-of-The-Art (Aircraft). Extra S.: Extra Supervision.

| Method                  | Backbone | Extra S. | Accuracy |
|-------------------------|----------|----------|----------|
| RCB [Wang et al. 2015] | VGGNet-16 | Yes      | 89.4     |
| MG-CNN [Wang et al. 2015] | VGGNet-19 | Yes      | 89.4     |
| KP [Cui et al. 2014] | VGGNet-16 | No       | 89.4     |
| LHBP [Rong and Fawkes 2017] | VGGNet-16 | No       | 89.4     |
| G"DNet [Wang, Li, and Zhang 2017] | VGGNet-16 | No       | 89.4     |
| Grassmann Pool [Wei et al. 2018] | VGGNet-16 | No       | 89.4     |
| HBP [Wu et al. 2018] | VGGNet-16 | No       | 89.4     |
| DFL-CNN [Wang, RoyChowdhury, and Maji 2015] | VGGNet-16 | No       | 89.4     |
| RACNN [Fu, Zheng, and Mei 2017] | VGGNet-16 | No       | 89.4     |
| MACNN [Zhang et al. 2017] | VGGNet-19 | No       | 89.4     |
| Deep KSSP [Rajagopal et al. 2018a] | VGGNet-19 | No       | 89.4     |
| STCNN [Liu et al. 2018b] | VGGNet-19 | No       | 89.4     |
| PC [Dabney et al. 2018a] | DenseNet-161 | No | 89.4     |
| Our API-Net | ResNet-101 | No | 89.4     |
| Our API-Net | DenseNet-161 | No | 89.4     |

4.2 Comparison with The-State-of-The-Art

We compare API-Net with a number of recent works on five widely-used benchmarks in Table 6. First, API-Net outperforms the object-part-localization approaches such as RACNN (Fu, Zheng, and Mei 2017) and MACNN (Zhang et al. 2017), showing the importance of API-Net for discriminative feature learning. Second, API-Net outperforms the patch-interaction approaches such as B-CNN (Lin, RoyChowdhury, and Maji 2015). HBP [Yu et al. 2018]. It demonstrates that, we need to put more efforts to distinguish two different images jointly, instead of learning with individual image alone. Third, API-Net outperforms the metric-learning approaches such as PC [Dabney et al. 2018a]. Particularly, we also re-implement TripletNet [Hoffer and Ailon 2013] in Table 6, a classical approach optimized by extra triplet loss, and compare it with API-Net. They all illustrate that, we should further exploit how to discover differential visual clues in the image pair, instead of straightforwardly regularizing the sample distribution in the feature space. Fourth, API-Net even outperforms the approaches with extra supervision such as localization annotations [Zhang et al. 2014], super-category labels [Chen et al. 2018a], or text [Li et al. 2018a]. It shows the effectiveness of API mod-
Finally, without complex architectures and multi-stage learning in the previous works, our API module can be easily integrated into the standard CNN training procedure and serve as a plug-and-play unit for discovering subtle visual differences. Hence, API-Net is a concise and practical deep learning approach for fine-grained classification.

### 4.3 Visualization

We further visualize API-Net in Fig. 3. First, we choose two pairs from the highly-confused categories in CUB-200-2011, i.e., Caspian Tern (CT) vs. Elegant Tern (ET), and Le Conte Sparrow (LCS) vs. Nelson Sharp Tailed Sparrow (NSTS). Second, we feed each pair into API-Net, and find top-5 activated channels of gate vectors $g_1$ and $g_2$. Subsequently, we show the corresponding feature maps ($14 \times 14$) before global pooling. Additionally, we show the corresponding feature maps in Baseline, i.e., ResNet-101 without attentive pairwise interaction.

As shown in red-dashed boxes of Fig. 3, many feature maps of Baseline are confused or noisy, e.g., the object regions are blurring, or certain background regions are activated. On the contrary, our API-Net can effectively discover distinct features, e.g., $g_1$ mainly focuses on the body of Caspian Tern, while $g_2$ mainly focuses on the mouth of Elegant Tern. These contrastive clues allow API to correctly distinguish such two birds.

Additionally, it is interesting to mention that, API-Net can automatically pay attention to discriminative object parts in the feature maps, even though it mainly works on high-level feature vectors. This observation indicates that, CNN can be well generalized via attentive pairwise interaction. As expected, our API-Net successfully recognizes all these pairs, while Baseline makes wrong predictions.

### 5 Conclusion

In this paper, we propose a novel Attentive Pairwise Interaction Network (API-Net) for fine-grained classification. It can adaptively discover contrastive cues from a pair of images, and attentively distinguish them via pairwise interaction. The results on five popular fine-grained benchmarks show that, API-Net achieves the state-of-the-art performance.

### 6 Acknowledgments

This work is partially supported by the National Key Research and Development Program of China (No. 2016YFC1400704), and National Natural Science Foundation of China (61876176, U1713208), Shenzhen Basic Research Program (JCYJ20170818164704758, CXB201104220032A), the Joint Lab of CAS-HK, Shenzhen Institute of Artificial Intelligence and Robotics for Society.
References

[Branson et al. 2014] Branson, S.; Van Horn, G.; Belongie, S.; and Perona, P. 2014. Bird species categorization using pose normalized deep convolutional nets. arXiv preprint arXiv:1406.2952.

[Bruner 2017] Bruner, J. 2017. A study of thinking. Routledge.

[Cai, Zuo, and Zhang 2017] Cai, S.; Zuo, W.; and Zhang, L. 2017. Higher-order integration of hierarchical convolutional activations for fine-grained visual categorization. In ICCV, 511–520.

[Chen et al. 2018] Chen, T.; Wu, W.; Gao, Y.; Dong, L.; Luo, X.; and Lin, L. 2018. Fine-grained representation learning and recognition by exploiting hierarchical semantic embedding. arXiv preprint arXiv:1808.04505.

[Cui et al. 2017] Cui, Y.; Zhou, F.; Wang, J.; Liu, X.; Lin, Y.; and Belongie, S. 2017. Kernel pooling for convolutional neural networks. In CVPR, 2921–2930.

[Dubey et al. 2018a] Dubey, A.; Gupta, O.; Guo, P.; Raskar, R.; Farrell, R.; and Naik, N. 2018a. Pairwise confusion for fine-grained visual classification. In ECCV, 70–86.

[Dubey et al. 2018b] Dubey, A.; Gupta, O.; Raskar, R.; and Naik, N. 2018b. Maximum-entropy fine grained classification. In NIPS.

[Engin et al. 2018] Engin, M.; Wang, L.; Zhou, L.; and Liu, X. 2018. Deepkspsd: learning kernel-matrix-based spd representation for fine-grained image recognition. In ECCV, 612–627.

[Fu, Zheng, and Mei 2017] Fu, J.; Zheng, H.; and Mei, T. 2017. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In CVPR, 4438–4446.

[Gao et al. 2016] Gao, Y.; Beijbom, O.; Zhang, N.; and Darrell, T. 2016. Compact bilinear pooling. In CVPR, 317–326.

[He et al. 2016] He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In CVPR, 770–778.

[Hermans, Beyer, and Leibe 2017] Hermans, A.; Beyer, L.; and Leibe, B. 2017. In defense of the triplet loss for person re-identification. arXiv preprint arXiv:1703.07737.

[Hoffer and Ailon 2015] Hoffer, E., and Ailon, N. 2015. Deep metric learning using triplet network. In International Workshop on Similarity-Based Pattern Recognition, 84–92. Springer.

[Huang et al. 2017] Huang, G.; Liu, Z.; Van Der Maaten, L.; and Weinberger, K. Q. 2017. Densely connected convolutional networks. In CVPR, 4700–4708.

[Jaderberg et al. 2015] Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial transformer networks. In NIPS, 2017–2025.

[Khosla et al. 2011] Khosla, A.; Jayadevaprakash, N.; Yao, B.; and Li, F.-F. 2011. Novel dataset for fine-grained image categorization: Stanford dogs. In CVPR Workshops.

[Kong and Fowlkes 2017] Kong, S., and Fowlkes, C. 2017. Low-rank bilinear pooling for fine-grained classification. In CVPR, 365–374.

[Krause et al. 2013] Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d object representations for fine-grained categorization. In ICCV Workshops, 554–561.

[Krause et al. 2015] Krause, J.; Jin, H.; Yang, J.; and Fei-Fei, L. 2015. Fine-grained recognition without part annotations. In CVPR, 5546–5555.

[Kulis and others 2013] Kulis, B., et al. 2013. Metric learning: A survey. Foundations and Trends® in Machine Learning 5(4):287–364.

[Li et al. 2018a] Li, J.; Zhu, L.; Huang, Z.; Lu, K.; and Zhao, J. 2018a. I read, i saw, i tell: texts assisted fine-grained visual classification. In ACM MMM, 663–671.

[Li et al. 2018b] Li, P.; Xie, J.; Wang, Q.; and Gao, Z. 2018b. Towards faster training of global covariance pooling networks by iterative matrix square root normalization. In CVPR, 947–955.

[Lin et al. 2015] Lin, D.; Shen, X.; Lu, C.; and Jia, J. 2015. Deep lac: Deep localization, alignment and classification for fine-grained recognition. In CVPR, 1666–1674.

[Lin, RoyChowdhury, and Maji 2015] Lin, T.-Y.; RoyChowdhury, A.; and Maji, S. 2015. Bilinear cnn models for fine-grained visual recognition. In ICCV, 1449–1457.

[Liu et al. 2016] Liu, X.; Xia, T.; Wang, J.; Yang, Y.; Zhou, F.; and Lin, Y. 2016. Fully convolutional attention networks for fine-grained recognition. arXiv preprint arXiv:1603.06765.

[Maji et al. 2013] Maji, S.; Rahtu, E.; Kannala, J.; Blaschko, M.; and Vedaldi, A. 2013. Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151.

[Schroff, Kalenichenko, and Philbin 2015] Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. Facenet: A unified embedding for face recognition and clustering. In CVPR, 815–823.

[Sun et al. 2018] Sun, M.; Yuan, Y.; Zhou, F.; and Ding, E. 2018. Multi-attention multi-class constraint for fine-grained image recognition. In ECCV, 805–821.

[Van Horn et al. 2015] Van Horn, G.; Branson, S.; Farrell, R.; Haber, S.; Barry, J.; Ipeirots, P.; Perona, P.; and Belongie, S. 2015. Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In CVPR, 595–604.

[Wah et al. 2011] Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset. California Institute of Technology.

[Wang et al. 2015] Wang, D.; Shen, Z.; Shao, J.; Zhang, W.; Xue, X.; and Zhang, Z. 2015. Multiple granularity descriptors for fine-grained categorization. In ICCV, 2399–2406.

[Wang et al. 2016] Wang, Y.; Choi, J.; Morariu, V.; and Davis, L. S. 2016. Mining discriminative triplets of patches for fine-grained classification. In CVPR, 1163–1172.

[Wang, Li, and Zhang 2017] Wang, Q.; Li, P.; and Zhang, L. 2017. G2denet: Global gaussian distribution embedding network and its application to visual recognition. In CVPR, 2730–2739.

[Wang, Morariu, and Davis 2018] Wang, Y.; Morariu, V. I.; and Davis, L. S. 2018. Learning a discriminative filter bank within a cnn for fine-grained recognition. In CVPR, 4148–4157.

[Wei et al. 2018] Wei, X.; Zhang, Y.; Gong, Y.; Zhang, J.; and Zheng, N. 2018. Grassmann pooling as compact homogeneous bilinear pooling for fine-grained visual classification. In ECCV, 355–370.

[Yang et al. 2018] Yang, Z.; Luo, T.; Wang, D.; Hu, Z.; Gao, J.; and Wang, L. 2018. Learning to navigate for fine-grained classification. In ECCV, 420–435.

[Yu et al. 2018] Yu, C.; Zhao, X.; Zheng, Q.; Zhang, P.; and You, X. 2018. Hierarchical bilinear pooling for fine-grained visual recognition. In ECCV, 574–589.

[Zhang et al. 2014] Zhang, N.; Donahue, J.; Girshick, R.; and Darrell, T. 2014. Part-based r-cnns for fine-grained category detection. In ECCV, 834–849. Springer.

[Zhang et al. 2016a] Zhang, X.; Zhou, F.; Lin, Y.; and Zhang, S. 2016a. Embedding label structures for fine-grained feature representation. In CVPR, 1114–1123.

[Zhang et al. 2016b] Zhang, X.; Xiong, H.; Zhou, W.; Lin, W.; and Perona, P. 2016b. Higher-order integration of hierarchical convolutional activations for fine-grained visual categorization. In ICCV, 511–520.
Tian, Q. 2016b. Picking deep filter responses for fine-grained image recognition. In CVPR, 1134–1142.

[Zhao et al. 2017] Zhao, B.; Wu, X.; Feng, J.; Peng, Q.; and Yan, S. 2017. Diversified visual attention networks for fine-grained object classification. TMM 19(6):1245–1256.

[Zheng et al. 2017] Zheng, H.; Fu, J.; Mei, T.; and Luo, J. 2017. Learning multi-attention convolutional neural network for fine-grained image recognition. In ICCV, 5209–5217.