Improving Fine-Grained Visual Recognition in Low Data Regimes via Self-Boosting Attention Mechanism

Yangyang Shu[1], Baosheng Yu[2], Haiming Xu[1], and Lingqiao Liu[1]⋆

[1] School of Computer Science, The University of Adelaide
{yangyang.shu,hai-ming.xu,lingqiao.liu}@adelaide.edu.au
[2] School of Computer Science, The University of Sydney
baosheng.yu@sydney.edu.au

Abstract. The challenge of fine-grained visual recognition often lies in discovering the key discriminative regions. While such regions can be automatically identified from a large-scale labeled dataset, a similar method might become less effective when only a few annotations are available. In low data regimes, a network often struggles to choose the correct regions for recognition and tends to overfit spurious correlated patterns from the training data. To tackle this issue, this paper proposes the self-boosting attention mechanism, a novel method for regularizing the network to focus on the key regions shared across samples and classes. Specifically, the proposed method first generates an attention map for each training image, highlighting the discriminative part for identifying the ground-truth object category. Then the generated attention maps are used as pseudo-annotations. The network is enforced to fit them as an auxiliary task. We call this approach the self-boosting attention mechanism (SAM). We also develop a variant by using SAM to create multiple attention maps to pool convolutional maps in a style of bilinear pooling, dubbed SAM-Bilinear. Through extensive experimental studies, we show that both methods can significantly improve fine-grained visual recognition performance on low data regimes and can be incorporated into existing network architectures. The source code is publicly available at: *https://github.com/GANPerf/SAM*.

Keywords: Self-boosting attention mechanism, fine-grained visual recognition, low data regimes

1 Introduction

Fine-Grained Visual Recognition (FGVR) aims to distinguish subcategories of objects under basic-level category, such as bird species [1,24], vehicle models [10,26], aircraft models [14]. The key challenge of FGVR is to discover the key object parts that can be used to identify object categories. In the existing

⋆ Corresponding author: lingqiao.liu@adelaide.edu.au. This work is supported by the Centre for Augmented Reasoning.

works, such a discovery is either explicitly achieved through part-mining [6,23] or implicitly learned in end-to-end training [15,32]. The latter strategy is the current state-of-the-art, which usually relies on special designs of the network, e.g., bilinear networks [12,4,9,29], to impose certain inductive bias.

Existing FGVR research is often based on a dataset with sufficient annotations, generally with more than 5,000 images and hundreds of categories. However, many practical FGVR problems do not have such a large dataset since annotating fine-grained data is a time-consuming, costly, and error-prone task. For example, labeling different bird species requires an expert in zoology. It remains unclear if the existing end-to-end learning methods can generalize well in the low data regime.

Unfortunately, from our empirical study (shown in section 5.3), we found that the existing solutions for FGVR may become less effective. It seems that when the number of training samples becomes smaller, the network tends to overfit the spurious patterns that happen to correlate with object categories. For example, when distinguishing different types of birds, the network may pay more attention to the surrounding environment rather than the bird body.

To overcome this issue, in this paper, we propose a novel solution called the self-boosting attention mechanism (SAM) to regularize the network to make the decision based on regions that are shared across instances and categories. Specifically, we first use existing visual explanation approaches such as CAM [30] and GradCAM [18] to obtain attention maps to highlight the key regions supporting the prediction of the ground-truth class. Then we use the generated attention maps as prediction targets and fit them with a class-agnostic projection from the convolutional feature map. In this way, we could encourage the network to use the features from the commonly attended regions to make a prediction. To further strengthen the regularization, we further developed a variant by using the above auxiliary task to regularize a set of projections, with each projection working as a part detector. Those projections allow us to leverage a bilinear pooling operation to obtain a new representation of the image. Through extensive experiments, we show that the proposed two strategies achieve superior performance than the competitive approaches for FGVR in a low data regime. Also, we demonstrate that the proposed method can be easily incorporated into the existing approach and achieve further performance boost.

2 Related Work

2.1 Fine-grained Visual Recognition

Locating distinctive regions plays an important and fundamental role in fine-grained visual recognition. In early researches, manually defined object and part annotations are extensively studied for fine-grained visual recognition. For example, Zhang *et al.*[28] use the trained R-CNN model to learn whole-object and part detectors with the help of part-level bounding boxes. Branson *et al.*[2] propose a method which is based on part detection. They use part and object bounding boxes to estimate a similarity-based warping function for improving the performance in fine-grained recognition tasks. However, manually defining object and part annotations requires additional human cost, largely limited in practical application. In the visual attention models community, Sermanet *et al.*[19] first propose to use attention models in FGVR. They use an attention-based RNN structure to direct high-resolution attention to the discriminative regions. However, the computational cost in their method is higher because they forward GoogLeNet three times. Xiao *et al.*[25] propose to use three types of attention in a deep neural network for the fine-grained classification task. The three types of attention are combined to train domain-specific deep nets: bottom-up attention, object-level top-down attention, and part-level top-down attention. Hu *et al*[7] use weakly supervised learning to generate attention maps only by image-level annotation. The generated attention maps in their proposed WS-DAN network ensure the model looks at the object better and closer. The main drawback is that attention models will be vulnerable and prone to over-fitting when the image-level annotation is quite a few.

Bilinear-based methods are very popular in fine-grained visual recognition. Lin *et al.*[12] first propose bilinear CNN models by two feature extractors to model local pairwise feature interactions for fine-grained visual recognition. Because original bilinear CNN models are high-dimensional and computationally expensive to train due to calculating pairwise interaction between channels, various studies of dimension reduction techniques have been proposed. Gao *et al.*[4] propose two compact bilinear pooling methods using two low-dimensional approximations of the polynomial kernel, Random Maclaurin[8] and Tensor Sketch [17] to generate the compact bilinear representations and reduce feature dimensions. Kong *et al.*[9] present a compact low-rank classification model and use the low-rank approximation to the covariance matrix to address the computational demands of high feature dimensionality. Zheng *et al*[29] propose a deep bilinear transformation (DBT) block to uniformly divide input channels into several semantic groups. The computational cost can be relieved via calculating pairwise interactions within each group. For our method, the feature dimensions can be reduced via controlling the number of the predicted attention maps when the element-wise multiplication.

2.2 Low-Supervised FGVR

To reduce the dependence on training data, some studies distinguish different categories with very little supervision, e.g., few-shot fine-grained visual recognition and semi-supervised learning for fine-grained visual recognition. Zhu *et al.*[31] propose a multi-attention meta-learning (MattML) method to capture discriminative parts of images for few-shot FGVR. The proposed MattML consists of the base learner and task learner, where the base learner is used for general feature learning, and the task learner uses a task embedding network to learn task representations. Wei *et al.*[22] proposed an end-to-end trainable deep network to solve few-shot FGVR. They use a bilinear feature learning module to capture the discriminative information of an exemplar image and use a classifier mapping module to map the intermediate feature into the decision boundary of the novel category. Lai *et al.*[11] propose an efficient method of semi-supervised learning, voted pseudo label (VPL), to improve the performance of classification in FGVR task when only a few samples are available. VPL is applied in unlabeled data to pick up their classes with non-confused labels, verified by the consensus prediction of different classification models. Mugnai *et al*[16] exploit semi-supervised learning to improve the performance of FGVR. They adopt an adversarial optimization strategy to combine the conditional entropy of unlabeled data with a second-order feature encoder to reduce the prohibitive annotation cost of FGVR. Our method works in a different setting to the above methods. Specifically, unlike semi-supervised FGVR, we do not assume the availability of unlabeled data; unlike few-shot FGVR, we do not require a relatively large amount of labeled samples from different categories as "base class samples".

3 Background

In this section, we briefly review Class Activation Maps (CAM) [30] and Gradient-weighted Class Activation Mapping (Grad-CAM) [18], which underpins the proposed Self-boosting Attention Mechanism.

3.1 Class Activation Maps

Class Activation Maps (CAM) are proposed to identify the importance of the image regions by projecting back the weights of the output layer onto the convolutional feature maps. CAM is applicable for the neural network architecture that uses Global Average Pooling (GAP) layer and classifier layers as the last two layers.

Let $\phi(I) \in \mathbb{R}^{H\times W\times D}$ represents the activation feature map of the last convolutional layer, where I is the input image and H, W and D are the height, width and the number of channels of the feature map, respectively. Thus the logits for class y, i.e., the decision value before the softmax, can be calculated as:

$$l(y) = \mathbf{w_y}^\top GAP(\phi(I)) = \mathbf{w_y}^\top \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}[\phi(I)]_{i,j} = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}\mathbf{w_y}^\top[\phi(I)]_{i,j}, \tag{1}$$

where $\mathbf{w_y}$ is the classifier for the y-th class. GAP represents Global Average Pooling and $[\phi(I)]_{i,j} \in \mathbb{R}^D$ denotes the feature vector located at the (i,j)-th grid. The class activation map (CAM) for class y is defined as:

$$[\mathrm{CAM}(y)]_{i,j} = \mathbf{w_y}^\top[\phi(I)]_{i,j}, \tag{2}$$

where $\mathrm{CAM}(y)$ denotes CAM for the y-th class. $[\mathrm{CAM}(y)]_{i,j}$ indicates the importance value of the (i,j)th spatial grid.

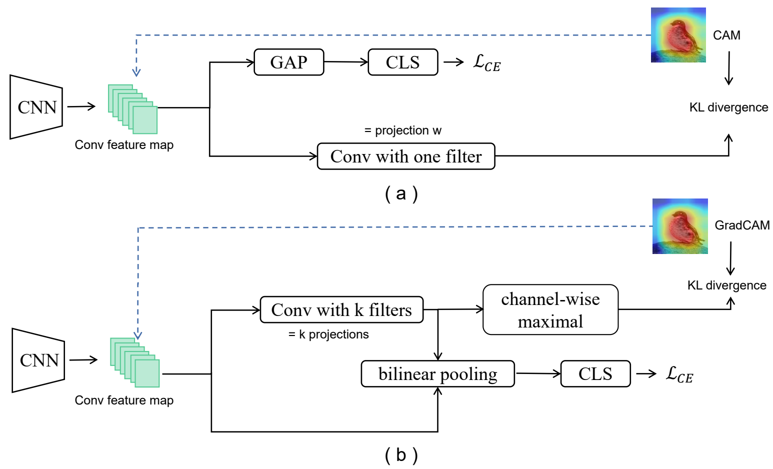

Fig. 1. The overview of our SAM network architecture. In the top half-section (a), the last convolutional layer feature maps in CNN network are used to obtain the cross-entropy loss function $\mathcal{L}_{CE}$ via global average pooling (GAP) and classifier (CLS). These feature maps via a linear projection are also enforced to fit the attention maps generated by CAM. The bottom half section (b) represents the method developed in bilinear pooling. The multiple projections are applied in convolutional feature maps to obtain multiple part detectors. Then a bilinear pooling operation is used to obtain a new feature representation. The feature maps after multi-projection and channel-wise maximal operation are also enforced to fit the attention maps generated by GradCAM.

3.2 Gradient-weighted Class Activation Mapping

Gradient-weighted class activation mapping (GradCAM) extends CAM by using the gradient information to calculate the importance of the activation. Unlike CAM, GradCAM could be applied to any convolutional layer or the input image, and is applicable to any neural network architecture. Formally, the GradCAM for the y-th class is calculated via:

$$[\text{Grad-CAM}(y)]_{i,j} = ReLU\left([\frac{\partial l(y)}{\partial[\phi(I)]_{i,j}}]^\top[\phi(I)]_{i,j}\right), \tag{3}$$

where Grad-CAM(y) denotes GradCAM for the y-th class. $[\text{Grad-CAM}(y)]_{i,j}$ refers to the importance value of the (i,j)th spatial grid. $l(y)$ is the logits for class y. $\frac{\partial l(y)}{\partial \phi(I)}$ is the gradient of the logits for class y w.r.t. the feature map $\phi(I)$.

4 Our Methods

4.1 Self-boosting Attention Mechanism

As introduced in the Introduction, the challenge of a fine-grained visual recognition system is to identify the key regions that can be discriminative for distinguishing the subtle difference across categories. With abundant training data and properly designed architecture, the key regions can usually be automatically learned via end-to-end training. However, as shown in our experiment (please see section 5.2 and 5.3), such an end-to-end training strategy becomes less effective in identifying the key regions when the number of training samples becomes smaller. In such a case, the spurious correlation and true discriminative patterns become hard to distinguish, and a network often mistakenly utilizes the former, leading to poor generalization.

To overcome this issue, this paper proposes a self-boosting strategy to regularize the network to encourage the use of regions shared across many instances and classes. Following the above notation, we hereafter use $\phi(I) \in \mathbb{R}^{H \times W \times D}$ to denote the last convolutional layer feature maps. The logits is obtained by applying a classifier $h_p(\cdot)$ to $\phi(I)$, that is, $p(y|I) = h_p(GAP(\phi(I)))$. The proposed regularization strategy constructs an auxiliary task for $\phi(I)$. Specifically, we first calculate CAM or GradCAM as attention maps[3] from $\phi(I)$ w.r.t the ground-truth class for each instance, denoting $g(I_n, y_n)$. Then we enforce $\phi(I)$ to fit $g(I_n, y_n)$ via a linear projection $\mathbf{w} \in \mathbb{R}^D$ without providing the ground-truth class information to the network, which could be implemented as applying a convolutional layer with a single filter. Specifically, we first normalize $g(I_n, y_n)$ and $\mathbf{w}^T\phi(I) \in \mathbb{R}^{H \times W}$ via the softmax function:

$$\begin{aligned} \bar{\mathbf{G}} &= \frac{\exp\left(g(I_n, y_n)/\tau\right)}{\sum_{i=1}^{H}\sum_{j=1}^{W}\exp\left([g(I_n, y_n)]_{i,j}/\tau\right)} \in \mathbb{R}^{H \times W}, \\ \bar{\mathbf{A}} &= \frac{\exp\left(\mathbf{w}^T\phi(I)/\tau\right)}{\sum_{i=1}^{H}\sum_{j=1}^{W}\exp\left([\mathbf{w}^T\phi(I)]_{i,j}/\tau\right)} \in \mathbb{R}^{H \times W}, \end{aligned} \tag{4}$$

where $[\cdot]_{i,j}$ denotes the i,j-th element of the feature map. τ is an empirical temperature parameter and we set it to 0.4 for all our experiments. For the simplistic of notations, we also slightly abuse the notation $\mathbf{w}^T\phi(I)$ to denote the feature map obtained by projecting the vector at each location of $\phi(I)$ through $\mathbf{w}$. This normalization will highlight the most important regions and we can also view the normalized feature maps are probability distributions. Then we can use Kullback-Leibler divergence, denoted as $\mathrm{KL}(\cdot,\cdot)$ to measure the compatibility between g' and p'. Thus the final loss function could be written as

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda\mathcal{L}_{SAM} = \mathcal{L}_{CE} + \lambda\mathrm{KL}(vec(\bar{\mathbf{A}}), vec(\bar{\mathbf{G}})). \tag{5}$$

At the first glance, the introduction of $\mathcal{L}_{SAM}$ seems to be slightly counter-intuitive. The attention map is generated from the current model, why allowing model to fit it will lead to any benefit? To understand its effect, one should notice that the attention map is calculated based on the ground-truth class. For example, if we use CAM to calculate the attention map, the CAM is calculated via $\mathbf{w}_{y_n}^\top[\phi(I_n)]_{i,j}$, that is, the classifier corresponding to the ground-truth class y_n is chosen to produce the CAM. In contrast, the projection vector $\mathbf{w}$ is class-agnostic. Thus, $\mathbf{w}^\top\phi(I)$ tends to fit the common part that are shared across all

[3] Once calculated, the attention map is detached from the back-propagation.

classes and instances. Also in this process, $\phi(I)$ will be learned to produce a good feature presentation for those common key parts. This in effect creates an inductive bias for encouraging the network to use the patterns from the common key parts to make prediction. We call this mechanism as self-boosting attention mechanism since the model will be boosted by fitting its own attention map. The illustration of this scheme can be seen in Figure 1 (a).

4.2 A Bilinear Pooling Extension of SAM

The rationale of the aforementioned SAM is that if $\phi(I)$ is learned to support detecting common parts via the auxiliary task, the network will also prefer to use the feature from the common parts to make a prediction. In such a design, we do not have hard constraints to enforce the network to only use features extracted from those common parts. In this section, we present an extension of SAM by explicitly introducing such a constraint.

The most straightforward approach is to use $A = \mathbf{w}^\top \phi(I)$ as an attention map to weight $\phi(I)$. In other words, instead of directly applying global average pooling to $\phi(I)$ to obtain image representation, we use the following attentive pooling scheme:

$$\mathbf{f} = \sum_{i,j} [A]_{i,j} [\phi(I)]_{i,j} \in \mathbb{R}^D. \tag{6}$$

With such a pooling scheme, the feature from un-attended regions, i.e. $[A]_{i,j} = 0$, will not be preserved into the pooled representation.

We notice that using attentive pooling is akin to the operation in bilinear pooling while the later is equivalent to using multiple attentions. Inspired by this analogy, we further create multiple attention maps $\{A_k | A_k = \mathbf{w}_k^\top \phi(I)\}, k = 1 \cdots K$ via multiple projections $\{\mathbf{w}_k\}$. Intuitively, we expect each attention map highlights one object part, and the union of them should fit the attention map calculated from GradCAM for the current image, as the latter showing all important regions contributing to the decision. In our method, we approximate the union of identified object key parts via taking the maximal value across all K attention maps, that is,

$$[A_u]_{i,j} = \max_k [A_k]_{i,j} \tag{7}$$

Then for each attention map, we can create a pooled feature via Eq. 6. We then concatenate the pooled features from all K feature maps as the final image representation:

$$\begin{aligned} \mathbf{f} &= cat(\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_K) \in \mathbb{R}^{DK} \\ \mathbf{f}_k &= \sum_{i,j} [A_k]_{i,j} [\phi(I)]_{i,j}, \end{aligned} \tag{8}$$

where $cat()$ denotes concatenation of vectors. Note that Eq. 8 is identical to the bilinear pooling [13]. So we call this extension SAM-Bilinear.

Table 1. Category and data splits on the `CUB-200-2011`, `Stanford Cars` and `FGVC-Aircraft` datasets

Datasets	Category	No. of training	No.of testing
`CUB-200-2011`	200	5994	5794
`Stanford Cars`	196	8144	8041
`FGVC-Aircraft`	100	6667	3333

The above idea can be implemented by following a bilinear neural network structure. The illustration of this scheme is shown in Figure 1 (b). To summarize, the network introduces a bilinear pooling module with one input being the last convolutional layer feature map $\phi(I)$ and the other input being K projections of $\mathbf{w}_k^\top \phi(I)$. This could be implemented by adding a convolutional layer with K filters after $\phi(I)$. The attention map is calculated by using GradCAM with respect to $\phi(I)$. Then a channel-wise max-pooling (Eq. 7) is applied after the added convolutional layer to obtain a predicted attention map. We then normalized both the predicted attention map and the generated attention map by following the scheme in Eq. 4.

5 Experiments

In this section, we conduct experiments to evaluate the performance of the SAM model for FGVR. The experimental conditions, including datasets, implementation Details etc. are firstly given in Section 5.1. In Section 5.2, ablation analysis are performed to investigate the effectiveness of each component in our model. We give the comparison with state-of-the-art methods in Section 5.3. In Section 5.4, we conduct experiments to exploit the effect of the number of line projections. Lastly, we use the visualization to explain our model in Section 5.5.

5.1 Experimental Conditions

Datasets In our experiments, we use three publicly available fine-grained visual datasets: `Caltech-UCSD Birds (CUB-200-2011)` [21], `Stanford Cars` [10] and `FGVC-Aircraft` [14]. The details of category and data splits in these three datasets are shown in Table 1. We reduce the number of labeled annotations, i.e., 10% to 50% for each category and the number of categories in our experiments to simulate the scenarios of low data regimes.

Implementation Details We implement our method using the PyTorch framework. In our experiments, the input images are resized to 256×256. Then a 224×224 patch is cropped randomly from the rescaled images on the three datasets for the purpose of data augmentation. ResNet-50 [5] is used as the architecture, and layer four is chosen as feature maps. The pre-trained weights of ResNet-50 on Imagenet are used for initialization. SGD optimizer with a mini-batch size of 24, weight decay of 1×10^{-4}, and a momentum of 0.9 are used to optimize the proposed network in our experiments. The learning rate of the classifier is 0.001. The parameter λ is 0.01.

Baselines We now compare our method to bilinear pooling-based methods. We choose the following four popular methods for comparison. For a fair comparison, we re-implement the method by changing VGG [20] with the ResNet framework.

- **Full Bilinear Pooling (FBP) [13]** uses an image as the input of two CNNs, and their outputs at each location are combined to obtain the bilinear feature representation. In [13], the $relu5\_3$ layer and $relu5$ layer truncated in a VGG-D [20] and VGG-M [3] networks respectively are used for obtaining bilinear. In this paper, we re-implement the method by following the identical structures as SAM-Bilinear for fair comparison. Specifically, we truncate at layer four of ResNet framework and apply K projections into the truncated layer four. The last convolutional feature maps and the outputs of projections are used to obtain the bilinear feature representation.
- **Compact Bilinear Pooling (CBP-TS) [4]** with Tensor Sketch projection is used in the same extract experimental setup as FBP. The projection dimension in [4] is found as $d = 8000$ to reach close-to maximum accuracy. We set d=500 to reach the maximum accuracy in our experiments.
- **Hierarchical Bilinear Pooling (HBP) [27]** integrates multiple cross-layer bilinear features to improve their representation capability. $relu5\_1$, $relu5\_2$ and $relu5\_3$ in VGG-16 in [27] are used because deeper layers contain more part semantic information. This paper applies HBP to ResNet network structures in the deeper layers (layer four). The dimension of joint embedding D in [27] is 8192*3. Given the computational complexity and classification performance, we set the same value as HBP in ResNet network structures.
- **Deep Bilinear Transformation (DBTNet-50) [29]** divides input channels into several semantic groups according to their semantic information and calculates pairwise interaction within semantic groups to obtain bilinear features efficiently. This also results in large saving in computation cost.

Experimental Design To meet the situation of a few annotations, we set the image-level annotations with four ratios, i.e., 10%, 15%, 30%, and 50%. Although only a few annotations are available, the proposed method employs a self-boosting attention mechanism to regularize the network and improve the classification performance of fine-grained tasks.

5.2 Main Results

To thoroughly investigate the proposed method, we conduct experiments to provide a detailed ablation analysis with different label proportions and categories on the three databases shown in Table 2. Our Resnet-50 method only uses the 2048D features representation extracted from the pre-trained ResNet-50 architecture without bilinear pooling and SAM operation. Our FBP is the method

Table 2. Evaluation of our method with four label proportions and four label categories on the `CUB200-2011` (Bird), `Stanford Cars` (Car) and `FGVC aircraft` (Aircraft) databases. SAM ResNet-50 applies the proposed SAM to the Resnet-50. Similarly, SAM-Bilinear combines the proposed SAM with FBP. (Bold numbers indicate the best performance. ↑ is the amount of increase compared to the respective baseline of SAM and SAM-Bilinear, i.e., ResNet-50 for SAM and FBP for SAM-Bilinear.)

Dataset	Category	Method	Label Proportion			
			10%	15%	30%	50%
Bird	30	ResNet-50	55.56%	61.55%	69.16%	77.65%
		SAM ResNet-50	58.76%$_{\uparrow 3.20\%}$	**65.79%**$_{\uparrow 4.24\%}$	70.16%$_{\uparrow 1.00\%}$	77.75%$_{\uparrow 0.10\%}$
		FBP	56.55%	61.93%	69.79%	77.86%
		SAM bilinear	**60.18%**$_{\uparrow 3.63\%}$	65.65%$_{\uparrow 3.72\%}$	**70.79%**$_{\uparrow 1.00\%}$	**78.28%**$_{\uparrow 0.42\%}$
	50	ResNet-50	45.72%	58.53%	68.47%	73.94%
		SAM ResNet-50	**51.14%**$_{\uparrow 5.42\%}$	61.80%$_{\uparrow 3.27\%}$	71.43%$_{\uparrow 2.76\%}$	75.19%$_{\uparrow 1.25\%}$
		FBP	42.82%	58.24%	68.67%	74.37%
		SAM bilinear	50.46%$_{\uparrow 7.64\%}$	**61.92%**$_{\uparrow 3.68\%}$	**72.07%**$_{\uparrow 3.40\%}$	**77.18%**$_{\uparrow 2.81\%}$
	100	ResNet-50	42.18%	56.28%	67.85%	75.39%
		SAM ResNet-50	46.27%$_{\uparrow 4.09\%}$	**60.16%**$_{\uparrow 3.88\%}$	70.82%$_{\uparrow\ 2.97\%}$	78.25%$_{\uparrow 2.86\%}$
		FBP	42.52%	55.94%	68.85%	75.71%
		SAM bilinear	**47.07%**$_{\uparrow 4.55\%}$	59.60%$_{\uparrow 3.66\%}$	**70.98%**$_{\uparrow 2.13\%}$	**78.53%**$_{\uparrow 2.82\%}$
	200	ResNet-50	36.99%	48.88%	62.60%	73.23%
		SAM ResNet-50	40.24%$_{\uparrow 3.25\%}$	52.05%$_{\uparrow 3.17\%}$	64.07%$_{\uparrow 1.47\%}$	73.92%$_{\uparrow 0.69\%}$
		FBP	37.88%	49.12%	63.27%	73.70%
		SAM bilinear	**41.83%**$_{\uparrow 3.95\%}$	**52.35%**$_{\uparrow 3.23\%}$	**65.19%**$_{\uparrow 1.92\%}$	**74.54%**$_{\uparrow 0.84\%}$
Car	30	ResNet-50	35.09%	45.72%	58.65%	68.53%
		SAM ResNet-50	39.95%$_{\uparrow 4.86\%}$	49.98%$_{\uparrow 4.26\%}$	61.90%$_{\uparrow 3.25\%}$	**75.86%**$_{\uparrow 2.33\%}$
		FBP	36.24%	46.14%	62.98%	73.92%
		SAM bilinear	**41.76%**$_{\uparrow 5.52\%}$	**50.49%**$_{\uparrow 4.35\%}$	**66.89%**$_{\uparrow 3.91\%}$	75.37%$_{\uparrow 1.43\%}$
	50	ResNet-50	34.38%	45.32%	62.64%	76.67%
		SAM ResNet-50	42.39%$_{\uparrow 8.01\%}$	**54.23%**$_{\uparrow 8.91\%}$	69.00%$_{\uparrow 6.36\%}$	79.14%$_{\uparrow 2.47\%}$
		FBP	37.76%	44.53%	63.43%	77.27%
		SAM bilinear	**43.23%**$_{\uparrow 5.47\%}$	54.18%$_{\uparrow 9.65\%}$	**69.15%**$_{\uparrow 5.72\%}$	**79.40%**$_{\uparrow 2.13\%}$
	100	ResNet-50	36.56%	47.46%	69.77%	79.86%
		SAM ResNet-50	47.42%$_{\uparrow 10.86\%}$	**59.18%**$_{\uparrow 11.72\%}$	75.75%$_{\uparrow 5.98\%}$	84.96%$_{\uparrow 5.10\%}$
		FBP	38.55%	50.32%	71.96%	81.51%
		SAM bilinear	**47.69%**$_{\uparrow 9.14\%}$	58.74%$_{\uparrow 8.42\%}$	**76.86%**$_{\uparrow 4.9\%}$	**85.23%**$_{\uparrow 3.72\%}$
	196	ResNet-50	37.45%	53.01%	75.26%	83.56%
		SAM ResNet-50	39.96%$_{\uparrow 2.51\%}$	55.02%$_{\uparrow 2.01\%}$	76.69%$_{\uparrow 1.43\%}$	84.85%$_{\uparrow 1.29\%}$
		FBP	40.13%	55.07%	76.42%	85.10%
		SAM bilinear	**43.19%**$_{\uparrow 3.06\%}$	**57.42%**$_{\uparrow 2.35\%}$	**77.63%**$_{\uparrow 1.21\%}$	**85.71%**$_{\uparrow 0.61\%}$
Aircraft	30	ResNet-50	26.70%	33.50%	47.00%	63.00%
		SAM ResNet-50	31.80%$_{\uparrow 5.10\%}$	37.70%$_{\uparrow 4.20\%}$	49.15%$_{\uparrow 2.15\%}$	65.10%$_{\uparrow 2.10\%}$
		FBP	26.90%	33.60%	46.70%	61.90%
		SAM bilinear	**32.50%**$_{\uparrow 5.60\%}$	**39.20%**$_{\uparrow 5.60\%}$	**51.80%**$_{\uparrow 5.10\%}$	**65.80%**$_{\uparrow 3.90\%}$
	50	ResNet-50	38.60%	45.20%	61.16%	70.29%
		SAM ResNet-50	43.58%$_{\uparrow 4.98\%}$	49.88%$_{\uparrow 4.68\%}$	63.79%$_{\uparrow 2.63\%}$	72.25%$_{\uparrow 1.96\%}$
		FBP	37.94%	45.44%	61.48%	71.79%
		SAM bilinear	**43.70%**$_{\uparrow 5.76\%}$	**50.84%**$_{\uparrow 5.40\%}$	**65.33%**$_{\uparrow 3.85\%}$	**72.95%**$_{\uparrow 1.16\%}$
	100	ResNet-50	43.52%	53.17%	71.32%	78.61%
		SAM ResNet-50	46.73%$_{\uparrow 2.21\%}$	56.02%$_{\uparrow 2.85\%}$	72.59%$_{\uparrow 1.27\%}$	79.21%$_{\uparrow 0.60\%}$
		FBP	45.16%	55.06%	72.12%	79.93%
		SAM bilinear	**47.97%**$_{\uparrow 2.81\%}$	**57.47%**$_{\uparrow 2.41\%}$	**73.43%**$_{\uparrow 1.31\%}$	**80.86%**$_{\uparrow 0.93\%}$

that uses bilinear pooling features representation. The method of SAM ResNet-50 uses the proposed self-boosting attention mechanism in ResNet-50, where the model does not use bilinear pooling features and only uses the last convolutional feature as the classifier's input. The method of SAM bilinear uses the proposed self-boosting attention mechanism in FBP.

From Table 2, we make the following observations: First, compared to the method of ResNet-50, SAM ResNet-50 increases the classification accuracy. For example, with the label proportion of 10%, 15%, 30% and 50% and the category of 200, the classification accuracy of SAM ResNet-50 are 40.24%, 52.05%, 64.07% and 73.92% respectively, which are 3.25%, 3.17%, 1.47% and 0.69% higher than ResNet-50 method on the `CUB200-2011` datasets. Similarly, significant improvement can also be found on the `Stanford Cars` and `FGVC Aircraft` datasets. This demonstrates the superiority of the proposed self-boosting attention mechanism. The model with a few label proportions is prone to overfit spurious correlated patterns. The proposed self-boosting attention mechanism regularizes the network and improves the classification performance in the testing set. A similar conclusion can also be found in comparing FBP and SAM bilinear. Second, compared to the method of ResNet-50, FBP has a better performance in most cases. This is because the method of FBP uses the bilinear pooling feature representation, which is more discriminative features for fine-grained visual recognition. Third, when the label proportion reduces from 50% to 10%, the gap performances between SAM ResNet-50/SAM bilinear and ResNet-50/FBP become larger. Fourth, with the category of label reduced from 200 to 30 on the `CUB200-2011` datasets, 196 to 30 on the `Stanford Cars` dataset, and 100 to 30 on the `FGVC Aircraft` dataset, respectively, there is generally a better improvement in the performance of the proposed model. The proposed self-boosting attention mechanism effectively regularizes the network and reduces over-fitting under smaller label proportions or category labels, which is more beneficial for fine-grained visual recognition.

5.3 Comparison with State-of-the-Art

We compare our method with state-of-the-art bilinear pooling methods on the `CUB200-2011`, `Stanford Cars` and `FGVC Aircraft` datasets shown in the Table 3. We can see that our SAM-based methods achieve state-of-the-art accuracy on the few label proportions on all these fine-grained datasets. Especially, we more significantly improve the classification accuracy on 10% and 15% label proportions compared to the improvement in 30%, 50% and 100% label proportions. We also incorporate the proposed SAM into the existing method of DBTNet-50 [29], which improves the performance when only a few annotations are available compared to the original DBTNet-50 method.

For the computational complexity, the bilinear feature dimensions in the method of FBP [13] in our experiments is $2048 * 16$. The method of CBP-TS [4] is proposed to reduce the bilinear feature dimension. Setting the reduced dimension as 500 in our experiments can achieve the best performance. The dimension

Table 3. Comparison with state-of-the-art FGVC methods with three label proportions on the three datasets. We also apply the proposed SAM to the state-of-the-art method DBTNet [29], creating a method dubbed SAM DBTNet-50, which shows compelling results with low feature dimension.

Dataset	Method	Dimension D	Label Proportion				
			10%	15%	30%	50%	100%
Bird	Fine-Tuning	2048	36.99%	48.88%	62.60%	73.23%	81.34%
	FBP [13]	2048*16	37.88%	49.12%	63.27%	73.70%	82.52%
	CBP-TS[4]	500	37.12%	47.82%	62.24%	72.37%	81.48%
	HBP [27]	$8192*\frac{n(n-1)}{2}$ †	38.57%	50.12%	63.86%	74.18%	86.12%
	DBTNet-50 [29]	2048	37.67%	49.52%	63.16%	73.28 %	86.04%
	SAM ResNet-50	2048	40.24%	52.05%	64.07%	73.92%	81.62%
	SAM DBTNet-50	2048	40.38%	52.02%	64.82%	74.12%	**87.26%**
	SAM bilinear	2048*16	**41.83%**	**52.35%**	**65.19%**	**74.54%**	81.86%
Car	Fine-Tuning	2048	37.45%	53.01%	75.26%	83.56%	91.02%
	FBP [13]	2048*16	40.13%	55.07%	76.42%	85.10%	91.63%
	CBP-TS[4]	500	37.77%	54.87%	75.51%	84.80%	89.52%
	HBP [27]	$8192*\frac{n(n-1)}{2}$ †	40.02%	55.82%	76.81%	85.31%	92.73%
	DBTNet-50 [29]	2048	39.48%	55.24%	76.52%	86.52%	**94.32%**
	SAM ResNet-50	2048	39.96	55.02%	76.69%	84.85%	91.06%
	SAM DBTNet-50	2048	42.47%	56.06%	**78.06%**	**86.86%**	94.18%
	SAM bilinear	2048*16	**43.19%**	**57.42%**	77.63%	85.71%	91.48%
Aircraft	Fine-Tuning	2048	43.52%	53.17%	71.32%	78.61%	87.13%
	FBP [13]	2048*16	45.16%	55.06%	72.12%	79.93%	87.32%
	CBP-TS[4]	500	44.63%	54.79%	71.32%	79.60%	84.58%
	HBP [27]	$8192*\frac{n(n-1)}{2}$ †	45.28%	56.12%	72.58%	81.47%	89.74%
	DBTNet-50 [29]	2048	45.35%	56.36%	73.06%	81.26%	90.86%
	SAM ResNet-50	2048	46.73%	56.02%	72.59%	79.21%	86.74%
	SAM DBTNet-50	2048	47.56%	**58.24%**	73.36%	**81.62%**	**91.18%**
	SAM bilinear	2048*16	**47.97%**	57.47%	**73.43%**	80.86%	87.46%

† n is the number of convolution layers features.

on the method of HBP [27] is $8192*\frac{n(n-1)}{2}$ where 8192 is the embedding dimension obtained by the project layer in [27], and n is the number of convolution layers features. We can find that with the increase of n, the dimension of the bilinear feature is higher. The dimension on the method of DBTNet-50 [29] is 2048 to keep feature dimensions unchanged. In our method, the proposed SAM can be used in DBTNet-50, demonstrating that SAM DBTNet-50 has a better performance when only a few annotations are available than the original DBTNet-50. In the proposed SAM bilinear, the dimension of bilinear pooling features in our method is $2048*16$, ensuring an acceptable dimension and high classification performance. In the proposed SAM ResNet-50, the feature dimension is unchanged and equal to 2048, resulting in large computation cost savings.

5.4 Analysis of the Number of Linear Projections in SAM-Bilinear

As elaborated in Section 4.2, we use multiple projections to leverage bilinear pooling operations to obtain a new representation of the image. It is also vital and can be used as part detectors to help the proposed network locate the object's discriminative part. To explore the impact of linear projections number K, we

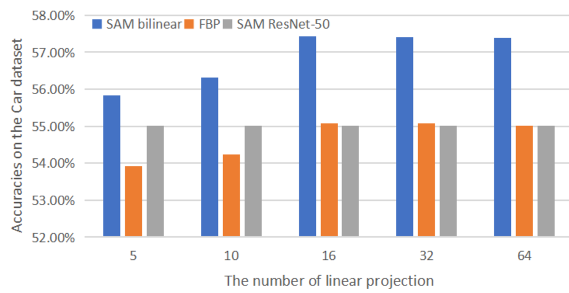

Fig. 2. Comparison of SAM bilinear, FBP and SAM ResNet-50 with different number of linear projections on the Stanford Cars dataset.

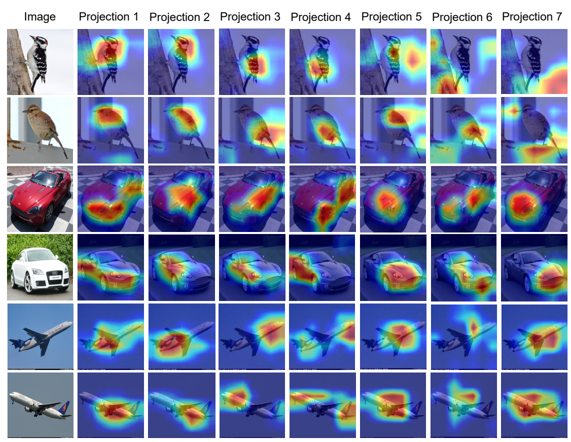

Fig. 3. Visualization of each part detector in the multi-projection on the three datasets. The first column is the original input images. The 2-8 columns are the visualization of the seven detected attention regions in seven linear projections.

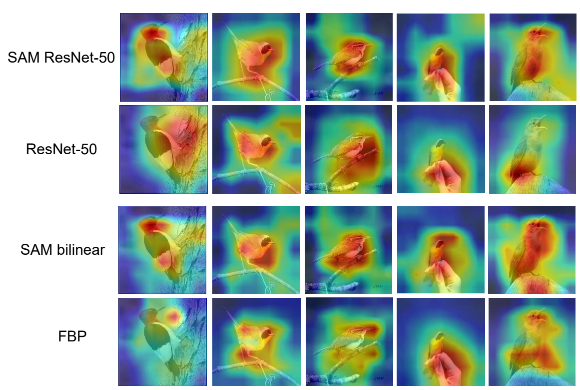

Fig. 4. Visualization of the attention regions in the method of SAM ResNet-50 vs ResNet-50 and SAM bilinear vs FBP with 15% label proportion.

conduct experiments on the proposed SAM bilinear, FBP and SAM ResNet-50 by setting the different numbers of linear projections. Take the Stanford Cars datasets with 15% label proportion, for example, our classification accuracy w.r.t. five different projections numbers are shown in Figure 2. From Figure 2, the accuracy significantly increases then gradually becomes stable in the method of SAM bilinear and FBP. The accuracy is peaked around 16, then slowly decrease with more heads (but only slightly). Please note that SAM ResNet-50 only needs one linear projection, and thus its accuracy is a constant in Figure 2.

5.5 Visualization

Visualization of Each Linear Projection The multi-projection has some practical implications. In our method, the number of linear projections is 16, and we visualize each result of linear projection under 15% label proportion and show them partly in Figure 3. As we can see, the highlighted regions of multi-projection reveal the significant parts that humans also rely on to improve the discriminative image representation, e.g., the head, body, and back for a bird, the head, tire and light for cars, and wings, head, and tail for aircraft.

Visualization of Attention Regions for SAM and SAM-Bilinear This visualization aims to explain why the proposed method is effective when the number of training data becomes small. We compare the method of ResNet-50 with the proposed SAM ResNet-50, FBP with the proposed SAM bilinear, and visualize their attention regions on the `CUB200-2011` dataset with 15% label proportions. From the visualization in Figure 4, we can see that the existing method may not attend to the correct regions when the number of training samples becomes small. In contrast, the proposed methods, either SAM or SAM-Bilinear can produce a more reasonable attention map. This indicates that the self-boosting attention mechanism can be used to correct the predicted attention regions when the number of training data becomes smaller, thus improving the performance of the fine-grained visual recognition task.

6 Conclusions

In this paper, we propose a self-boosting attention mechanism (SAM) for fine-grained visual recognition to regularize the network with low data regimes. The proposed SAM enforces the network to focus on the key regions shared across samples and classes. These key regions are constrained to fit the attention maps generated from CAM/GradCAM. Unlike previous work identifying the key regions that rely on abundant training data, our self-boosting attention mechanism is still effective when the number of training samples becomes smaller. Furthermore, we extend the proposed SAM with the bilinear model to further strengthen the regularization. The proposed SAM effectively regularize the network when image-level annotations are quite a few, and outperforms existing state-of-the-art on the `CUB200-2011`, `Stanford Cars` and `FGVC Aircraft` datasets.

References

1. Berg, T., Liu, J., Woo Lee, S., Alexander, M.L., Jacobs, D.W., Belhumeur, P.N.: Birdsnap: Large-scale fine-grained visual categorization of birds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2011–2018 (2014)
2. Branson, S., Van Horn, G., Belongie, S., Perona, P.: Bird species categorization using pose normalized deep convolutional nets. arXiv preprint arXiv:1406.2952 (2014)
3. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. arXiv preprint arXiv:1405.3531 (2014)
4. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact bilinear pooling. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 317–326 (2016)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
6. He, X., Peng, Y.: Weakly supervised learning of part selection model with spatial constraints for fine-grained image classification. In: Thirty-first AAAI conference on artificial intelligence (2017)
7. Hu, T., Qi, H., Huang, Q., Lu, Y.: See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification. arXiv preprint arXiv:1901.09891 (2019)
8. Kar, P., Karnick, H.: Random feature maps for dot product kernels. In: Artificial intelligence and statistics. pp. 583–591. PMLR (2012)
9. Kong, S., Fowlkes, C.: Low-rank bilinear pooling for fine-grained classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 365–374 (2017)
10. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 554–561 (2013)
11. Lai, D., Tian, W., Chen, L.: Improving classification with semi-supervised and fine-grained learning. Pattern Recognition **88**, 547–556 (2019)
12. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear cnn models for fine-grained visual recognition. In: Proceedings of the IEEE international conference on computer vision. pp. 1449–1457 (2015)
13. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear convolutional neural networks for fine-grained visual recognition. IEEE transactions on pattern analysis and machine intelligence **40**(6), 1309–1322 (2017)
14. Maji, S., Rahtu, E., Kannala, J., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151 (2013)
15. Min, S., Yao, H., Xie, H., Zha, Z.J., Zhang, Y.: Multi-objective matrix normalization for fine-grained visual recognition. IEEE Transactions on Image Processing **29**, 4996–5009 (2020)
16. Mugnai, D., Pernici, F., Turchini, F., Del Bimbo, A.: Fine-grained adversarial semi-supervised learning. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM) **18**(1s), 1–19 (2022)
17. Pham, N., Pagh, R.: Fast and scalable polynomial kernels via explicit feature maps. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 239–247 (2013)

18. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. pp. 618–626 (2017)
19. Sermanet, P., Frome, A., Real, E.: Attention for fine-grained categorization. arXiv preprint arXiv:1412.7054 (2014)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
21. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)
22. Wei, X.S., Wang, P., Liu, L., Shen, C., Wu, J.: Piecewise classifier mappings: Learning fine-grained learners for novel categories with few examples. IEEE Transactions on Image Processing **28**(12), 6116–6125 (2019)
23. Wei, X.S., Xie, C.W., Wu, J., Shen, C.: Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization. Pattern Recognition **76**, 704–714 (2018)
24. Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P.: Caltech-ucsd birds 200. technical report cns-tr-2010-001. California Institute of Technology (2010)
25. Xiao, T., Xu, Y., Yang, K., Zhang, J., Peng, Y., Zhang, Z.: The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 842–850 (2015)
26. Yang, L., Luo, P., Change Loy, C., Tang, X.: A large-scale car dataset for fine-grained categorization and verification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3973–3981 (2015)
27. Yu, C., Zhao, X., Zheng, Q., Zhang, P., You, X.: Hierarchical bilinear pooling for fine-grained visual recognition. In: Proceedings of the European conference on computer vision (ECCV). pp. 574–589 (2018)
28. Zhang, N., Donahue, J., Girshick, R., Darrell, T.: Part-based r-cnns for fine-grained category detection. In: European conference on computer vision. pp. 834–849. Springer (2014)
29. Zheng, H., Fu, J., Zha, Z.J., Luo, J.: Learning deep bilinear transformation for fine-grained image representation. Advances in Neural Information Processing Systems **32** (2019)
30. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2921–2929 (2016)
31. Zhu, Y., Liu, C., Jiang, S.: Multi-attention meta learning for few-shot fine-grained image recognition. In: IJCAI. pp. 1090–1096 (2020)
32. Zhuang, P., Wang, Y., Qiao, Y.: Learning attentive pairwise interaction for fine-grained classification. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 13130–13137 (2020)