Learning Common Rationale to Improve Self-Supervised Representation for Fine-Grained Visual Recognition Problems

Yangyang Shu Anton van den Hengel Lingqiao Liu*

School of Computer Science, The University of Adelaide
{yangyang.shu,anton.vandenhengel,lingqiao.liu}@adelaide.edu.au

Abstract

Self-supervised learning (SSL) strategies have demonstrated remarkable performance in various recognition tasks. However, both our preliminary investigation and recent studies suggest that they may be less effective in learning representations for fine-grained visual recognition (FGVR) since many features helpful for optimizing SSL objectives are not suitable for characterizing the subtle differences in FGVR. To overcome this issue, we propose learning an additional screening mechanism to identify discriminative clues commonly seen across instances and classes, dubbed as common rationales in this paper. Intuitively, common rationales tend to correspond to the discriminative patterns from the key parts of foreground objects. We show that a common rationale detector can be learned by simply exploiting the GradCAM induced from the SSL objective without using any pre-trained object parts or saliency detectors, making it seamlessly to be integrated with the existing SSL process. Specifically, we fit the GradCAM with a branch with limited fitting capacity, which allows the branch to capture the common rationales and discard the less common discriminative patterns. At the test stage, the branch generates a set of spatial weights to selectively aggregate features representing an instance. Extensive experimental results on four visual tasks demonstrate that the proposed method can lead to a significant improvement in different evaluation settings.[1]

1. Introduction

Recently, self-supervised representations Learning (SSL) has been shown to be effective for transferring the learned representations to different downstream tasks [2, 4, 5, 14, 18]. Methods such as contrastive learning [7, 9, 16, 18] have demonstrated state-of-the-art feature learning capability and have been intensively studied recently. However, recent studies [11] suggest that contrastive learning may have a "coarse-grained bias" and could be less effective for fine-grained classification problems whose goal is to distinguish visually similar subcategories of objects under the basic-level category.

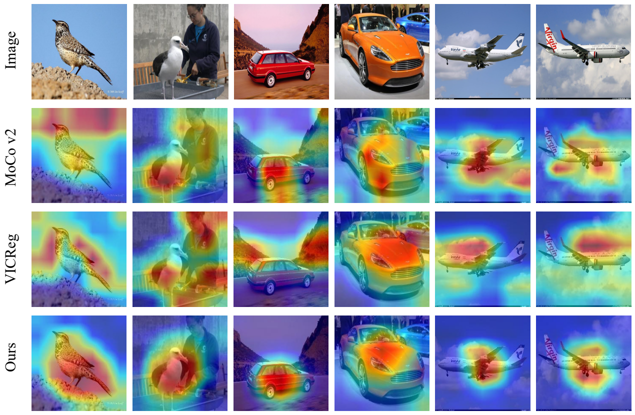

Figure 1. GradCAM [30] visualization for MoCo v2 [18], VICReg [4] and our method on the `CUB-200-2011`, `Stanford Cars` and `FGVC Aircraft` datasets. Compared with the existing method MoCo v2 and VICReg, which are prone to be distracted by background patterns, our method can identify features from the foreground and potentially the key parts of the object.

This phenomenon is rooted in fine-grained visual recognition (FGVR) properties and the training objective of SSL. SSL tries to minimize a pretext task, e.g., contrastive learning minimizes the distance between same-instance features while maximizing the distance among different-instance features, and any visual patterns that could contribute to loss minimization will be learned. On the other hand, the discriminative patterns for FGVR can be subtle. It is more likely to reside on key object parts. Thus, the feature learned

[*] Corresponding author. This work is supported by the Centre for Augmented Reasoning.

[1] The source code will be publicly available at: https://github.com/GANPerf/LCR

from SSL may not necessarily be useful for FGVR. Figure 1 shows our investigation of this issue. As seen, the existing SSL paradigms, such as MoCo [18] and VICReg [4] are prone to learn patterns from irrelevant regions[2]. Existing works [31, 38, 42] usually handle this issue by recoursing to pre-trained object part detectors or saliency detectors to regularize SSL using patterns from valid regions to achieve the training objective. However, both the part detectors and saliency detectors could restrict the applicability of SSL for FGVR since part detectors are trained from human annotations, and the saliency regions are not always coincident with the discriminative regions.

Therefore, this work aims to directly solve the problem from the target domain data. Specifically, we propose to learn an additional screening mechanism in addition to the contrastive learning process. The purpose of the screening mechanism is to filter out the patterns that might contribute to SSL objectives but are irrelevant to FGVR. Somehow surprisingly, we find that such a screening mechanism can be learned from the GradCAM [30] of the SSL loss via an extremely simple method. The whole process can be described as a "fitting and masking" procedure: At the training time, we use an additional branch with limited fitting capacity (see more discussion about it in Section 3.2) to fit the GradCAM calculated from the SSL objective. At the testing time, we apply this additional branch to predict an attention mask to perform weighted average pooling for the final presentation. The motivation for such a design is that the GradCAM fitting branch tends to characterize the discriminative patterns that commonly occur across samples due to its limited fitting capacity, and those common patterns, dubbed common rationale in this paper, are more likely corresponding to the discriminative clues from key object parts or at least foreground regions.

We implement our method based on MoCo v2 [8], one of the state-of-the-art approaches in unsupervised feature learning, which also produces the best performance on FGVR in our setting. Our implementation uses the training objective of MoCo v2 to learn feature representations and derive the GradCAM. Through extensive experiments, we show that our approach can significantly boost the quality of the learned feature. For example, when evaluating the learned feature for retrieval tasks, our method achieves 49.69% on the `CUB-200-2011` retrieval task, which is 32.62% higher than our baseline method (MoCo v2 [8]). In the linear evaluation phase, the proposed method achieves new state-of-the-art 71.31% Top-1 accuracy, which is 3.01% higher than MoCo v2.

2. Related Work

Self-supervised learning aims to learn feature representation from unlabeled data. It relies on a pretext task whose supervision could be derived from the data itself, e.g., image colorization [41], image inpainting [28], and rotation [24] prediction. Contrastive learning is recently identified as a promising framework for self-supervised learning and has been extensively studied [5, 7, 10, 13, 18, 36]. Despite the subtle differences, most contrastive learning approaches [15, 37, 43] try to minimize the distance between different views of the same images and push away the views of different images. The representative methods are SimCLR [7] and MoCo [18]. Besides contrastively learning, consistency-based approaches, such as BYOL [16], SimSiam [9] and masking-and-prediction-based approaches, such as MAE [17], BEiT [3], and ADIOS [32] are also proven effective for SSL.

Improving SSL via Better Region Localization. A pipeline to improve the distinguishing ability is to design better data augmentations of SSL. Three such methods were recently proposed: DiLo [42], SAGA [39], and ContrastiveCrop [29]. DiLo uses a copy-and-pasting approach as a kind of augmentation to create an image with a different background. In such a way, the proposed method distills localization via learning invariance against backgrounds and improves the ability of SSL models to localize foreground objects. SAGA adopts self-augmentation with a guided attention strategy to augment input images based on predictive attention. In their method, an attention-guided crop is used to enhance the robustness of feature representation. ContrastiveCrop shows a better crop as an augmentation to generate better views in SSL and keep the semantic information of an image. All of these works locate the object by improving the data augmentations for SSL. In this work, our method is adaptive to locate the key regions for self-supervised learning without needing external augmentations. Another family of approaches tries to target the same problem as ours: making the learned feature capture more of the foreground region. CVSA [38] proposed a cross-view saliency alignment framework that first crops and swaps saliency regions of images as a novel view generation. Then it adopts a cross-view saliency alignment loss to encourage the model to learn features from foreground objects. CAST [31] encourages Grad-CAM attention to fit the salient image regions detected by a saliency detector. Those methods often rely on pre-trained saliency detection. This implicitly assumes that salient regions are more likely to be discriminative regions. This assumption, however, does not always hold, especially for FGVR.

[2] The weakness of existing SSL methods on FGVR can also be quantitively demonstrated by their performance on retrieval tasks shown in Section 4.3. As will be seen from our experiments, using only the features learned from SSL and without any further supervision from the target data, existing SSL methods achieve very poor performance in the retrieval task, i.e., based on feature similarity to identify same-class images.

3. Method

In this section, we first briefly review self-supervised contrastive learning [8] and gradient-weighted class activation mapping (GradCAM) [30] as preliminary knowledge. Then, we introduce the proposed approach.

3.1. Preliminary

Self-Supervised Contrastive Learning. Given an image I from a batch of samples, $x = t(I)$ and $x' = t'(I)$ are the two augmented views, where the t and t' are two different transformations sampled from a set of data augmentations $\mathcal{T}$. Then the views x and x' are used as the input of an encoder f_θ to generate their feature representations $u = f_\theta(x)$ and $u' = f_\theta(x')$. Finally, the projector heads g_θ are used to map u and u' onto the embeddings $z = g_\theta(u)$ and $z' = g_\theta(u')$. Let (z, z') be embeddings from the same image and are used as a positive pair. Let z_k be the embedding from a different image, and (z, z_k) thus composes a negative pair. SimCLR [7] adopts a contrastive loss to maximize the agreement of positive pairs over those of negative pairs. The MoCo [1, 10, 18] family adopts the same contrastive loss but adds a queue to store the image embeddings to alleviate the memory cost due to the large batch size. Formally, the contrastive loss takes the following form

$$\mathcal{L}_{CL} = -\log \frac{\exp(z \cdot z'/t)}{\sum_{i=0}^{Q} \exp(z \cdot z_i/t)}, \tag{1}$$

where t denotes a temperature parameter, and z_i is the embedding in the queue.

Gradient-weighted Class Activation Mapping (GradCAM) is a commonly used way to produce a visual explanation. It identifies the important image regions contributing to the prediction by using the gradient of the loss function with respect to the feature map or input images. In this paper, we consider the gradient calculated with respect to the last convolutional layer feature maps. Formally, we consider the feature map of the last convolutional layer denoting as $\phi(I) \in \mathbb{R}^{H \times W \times C}$, H, W and C are the height, width and number of channels of the feature map, respectively. In standard C-way classification, the GradCAM is calculated by:

$$\begin{aligned} &[\text{Grad-CAM}(\hat{y})]_{i,j} = ReLU\left(\alpha_{\hat{y}}^{\top}[\phi(I)]_{i,j}\right) \\ &where,\ \alpha_{\hat{y}} = \frac{\partial \mathcal{L}_{CE}(P(y), \hat{y})}{\partial [\phi(I)]_{i,j}} \in \mathbb{R}^{C}, \end{aligned} \tag{2}$$

where $\mathcal{L}_{CE}(P(y), \hat{y})$ is the cross-entropy loss measuring the compatibility between the posterior probability $P(y)$ and ground-truth class label $\hat{y}$[3]. $[\phi(I)]_{i,j} \in \mathbb{R}^{C}$ denotes the feature vector located at the (i, j)-th grid. Grad-CAM$(\hat{y})$ denotes GradCAM for the $\hat{y}$-th class. $[\text{Grad-CAM}(\hat{y})]_{i,j}$ refers to the importance value of the (i, j)th spatial grid for predicting the $\hat{y}$-th class.

Note that although GradCAM is commonly used for supervised classification problems, it can be readily extended to other problems by changing the corresponding loss function.

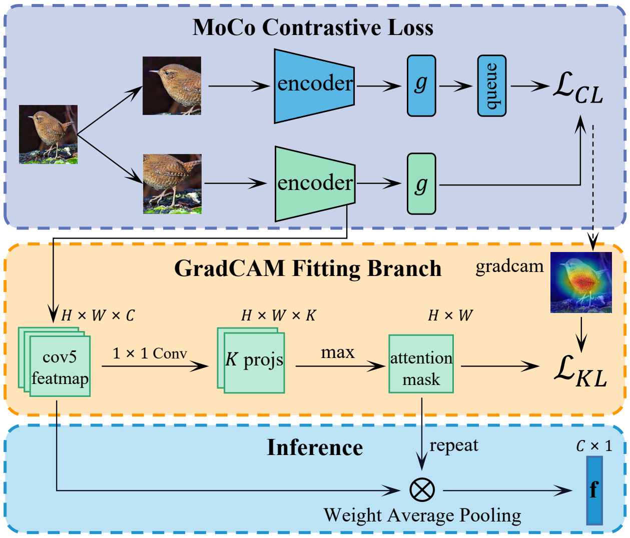

Figure 2. The overview of our method. At the structure level, it has the following components: 1)*Encoder*: MoCo-based contrastive learning is used to create the image-level representation. 2) *GradCAM fitting branch*, which is a convolutional layer with k filters followed by a channel-wise max-out operator. 3) *Inference*: at the inference time, the prediction from GFB is used as a spatial attention mask, and it is used to replace the GAP operation by weighted average pooling to obtain the image representation.

3.2. Our Method: Learning Common Rationale

Figure 2 gives an overview of the proposed method. Our method is extremely simple: we merely add one GradCAM fitting branch (GFB) and fit the GradCAM generated from the CL loss at the training time. At the test time, we let this GFB predict (normalized) GradCAM and use the prediction as an attention mask to perform weighted average pooling over the convolutional feature map. The details about this framework are elaborated as follows.

GradCAM Calculation. For self-supervised learning, we do not have access to ground-truth class labels. Therefore, we use the contrastive learning loss $\mathcal{L}_{CL}$ in Eq. 1 to replace $\mathcal{L}_{CE}$ in Eq. 2:

$$[\mathbf{G}]_{i,j} = ReLU\left(\frac{\partial \mathcal{L}_{CL}(\psi(\phi(I)))}{\partial [\phi(I)]_{i,j}}^{\top}[\phi(I)]_{i,j}\right), \tag{3}$$

where $\psi()$ denotes the feature extractor. Note that in the

[3] $\mathcal{L}_{CE}(P(y), \hat{y}) = \log P(\hat{y})$ for the multi-classification problem and the gradient of $\log P(y)$ is proportional to the gradient of the corresponding logit. Those equivalence forms lead to the different definitions of GradCAM in the literature.

original GradCAM, the GradCAM weight indicates how important each region contributes to classifying the image into the y-th class. Similarly, the GradCAM weight from $\mathcal{L}_{CL}$ indicates the contribution of each region to the instance discrimination task in the contrastive learning objective. From Figure 1, we can see that existing Contrastive Learning approaches learn a diverse set of visual patterns, and not all of them are relevant to the FGVR task.

Architecture of the GFB. The structure of the GFB plays an important role in our method. We expect this branch has a limited fitting capability, such that the branch will not overfit the GradCAM but only capture the commonly occurring discriminative patterns, i.e., common rationales. Inspired by [33], we use a convolutional layer with K filter and 1×1 kernel size followed by the max-out operation [33] as the fitting branch. Formally, such a branch applies the following operation to the local feature $[\phi(I)]_{i,j}$at the (i,j)-th grid of the feature map:

$$A_{i,j} = \max_k \{\mathbf{w}_k^\top [\phi(I)]_{i,j}\}, \tag{4}$$

where $A_{i,j}$ will be the predicted GradCAM weight at the (i,j)-th grid and $\mathbf{w}_k$ is the k-th filter (projector) in the convolutional layer. Intuitively, the convolutional layer can be seen as a collection of K detectors and the above operation can be understood as follows: each projection vector $\mathbf{w}_k$ detects an object part $\mathcal{P}_k$; the max-out operation takes the maximum of K projections at a given location, which will result in a high value if *any of the K object parts are detected*. Varying K could adjust the size of the detector pool and influence the fitting capability.

Training Loss. In addition to the contrastive learning loss, we require the GFB to produce a similar attention map as the one produced from the GradCAM of $\mathcal{L}_{CL}$. We follow [33] to normalize the GradCAM into a probability distribution and adopt KL divergence as the loss function.

Formally, we normalize the GradCAM $\mathcal{G}$ and $\mathcal{A}$ via the softmax function:

$$[\bar{\mathbf{G}}]_{i,j} = \frac{\exp\left([\mathbf{G}]_{i,j}/\tau\right)}{\sum_{i=1}^{H}\sum_{j=1}^{W}\exp\left([\mathbf{G}]_{i,j}/\tau\right)},$$
$$[\bar{\mathbf{A}}]_{i,j} = \frac{\exp\left([\mathbf{A}]_{i,j}/\tau\right)}{\sum_{i=1}^{H}\sum_{j=1}^{W}\exp\left([\mathbf{A}]_{i,j}/\tau\right)}, \tag{5}$$

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 here. Thus, the objective can be expressed as follows:

$$\mathcal{L}_{KL} = \sum_{i=1}^{H}\sum_{j=1}^{W}[\bar{\mathbf{A}}]_{i,j}\log\frac{[\bar{\mathbf{A}}]_{i,j}}{[\bar{\mathbf{G}}]_{i,j}}, \tag{6}$$

Thus, the final loss function taken on all images over an unlabelled dataset is shown as followings.

$$\mathcal{L} = \lambda\mathcal{L}_{CL} + \nu\mathcal{L}_{KL}. \tag{7}$$

Inference. At the inference time, the GradCAM fitting branch will be firstly used to produce an attention mask, i.e., prediction of GradCAM. This attention mask is firstly normalized using $A'_{i,j} = \frac{A_{i,j}-\min(A)}{1e^{-7}+\max(A)}$. Then we use the normalized attention to perform weighted average pooling of features from the last-layer convolutional feature map. Formally, it calculates:

$$\mathbf{f} = \sum_{i,j}[A']_{i,j}[\phi(I)]_{i,j} \in \mathbb{R}^C. \tag{8}$$

$\mathbf{f}$ is used for the downstream tasks.

Discussion. Our approach is inspired by the SAM method in [33]. However, there are several important differences:

- SAM method was proposed for supervised FGVC under a low-data regime, while our approach is designed for self-supervised learning.

- Most importantly, our study discovers that for self-supervised feature learning, the best strategy to interact with the GFB and the feature learning branch is different from what has been discovered in [33]. Table 1 summarizes the main differences. As seen, unlike either SAM or SAM-bilinear, we do not introduce any interaction between the feature learning branch and the GradCAM fitting branch during training but allow them to perform weighted average pooling at the test stage. In fact, we find that applying cross-branch interaction at the training stage will undermine the feature learning process. This is because applying cross-branch interaction, e.g., using $\mathbf{A}$ to weighted pool features will prevent CL from exploring discriminative patterns, especially when $\mathbf{A}$ has not been properly learned.

- For multiple projections, SAM uses bilinear pooling to aggregate $\mathbf{A}$ and the original feature map, resulting in high-dimensional feature representation. Our work performs max-out on multiple projections, resulting in a single attention mask for performing weighted average pooling. Consequently, we could achieve a significant reduction of the feature dimensionality.

To make our study comprehensive, we also explore several variants as baseline approaches. From the comparison with those methods, the benefit of our design could become more evident. For comparison, we use the following architectures as baselines for self-supervised pre-training:

SAM-SSL: this baseline extends SAM by changing its objective function from cross-entropy to the contrastive loss in MoCo V2. It shares the same architecture as SAM [33], where a projection is used as the GFB, and GradCAM fitting is trained as an auxiliary task to contrastive learning.

SAM-SSL-Bilinear: this baseline extends SAM-bilinear by using the contrastive loss in MoCo V2. The cross-branch

Table 1. Upper part: summary of the major difference between our method and SAM method [33]. Lower part: two variants that are also investigated in this work.

Method	GradCAM Fitting Branch	Cross-branch Interaction		Feature Dimension	Loss Function
		train	test		
SAM	1 proj	×	×	C	CrossEntropy
SAM-Bilinear	max[K projs]	bilinear pooling	bilinear pooling	C*K	CrossEntropy
Ours	max[K projs]	×	weighted average pooling	C	Contrastive
Ours-DualPooling	max[K projs]	weighted average pooling	weighted average pooling	C	Contrastive
Ours-MultiTask	max[K projs]	×	×	C	Contrastive

interaction of *SAM-SSL-Bilinear* follows the original SAM-Bilinear method.

Besides the above two extensions of the SAM methods, we also consider two variants of our method. The first is called *Ours-MultiTask*, which does not perform weighted average pooling at the test stage but merely uses Grad-CAM fitting as an auxiliary task. Another is called *Ours-DualPooling*, which performs weighted average pooling at both training and testing stages. Those two variants are summarized in Table 1.

4. Experiments

In this section, we will evaluate our proposed method on three widely used fine-grained visual datasets (`Caltech-UCSD Birds (CUB-200-2011)` [35], `Stanford Cars` [25] and `FGVC-Aircraft` [27]), and a large-scale fine-grained dataset (`iNaturalist2019` [34]). Our experiments aim to understand the effectiveness and the components of the proposed algorithm.

4.1. Datasets and Settings

Datasets. `CUB-200-2011` contains 11,788 images with 200 bird species, where 5994 images are used for training and 5794 images for testing. `Stanford Cars` contains 16,185 images with 196 categories, where 8144 images are for training and 8041 images for testing. `FGVC-Aircraft` contains 10,000 images with 100 categories, where 6667 images are for training, and 3333 images are for testing. `iNaturalist2019` in its 2019 version contains 1,010 categories, with a combined training and validation set of 268,243 images. Note that fine-grained visual task focus on distinguishing similar subcategories within a super-category, while there are six super-categories in `iNaturalist2019`.

Implementation Details. We adopt the ResNet-50 [19] as the network backbone, which is initialized using ImageNet-trained weights, and build our method on top of MoCo v2 [8]. Therefore the SSL loss term is identical to MoCo v2. The momentum value and memory size are set similarly to MoCo v2, i.e., 0.999 and 65536, respectively. The projector head g_θ in MoCo v2 is composed of two fully-connected layers with ReLU and a third linear layer with batch normalization (BN) [21]. The size of all three layers is 2048 ×2048×256. We set the mini-batch size as 128 and used an SGD optimizer with a learning rate of 0.03, a momentum of 0.9, and a weight decay of 0.0001. 100 epochs are used to train the feature extractor. The images from the four FGVR datasets are resized to 224×224 pixels during training times. During testing time, images are firstly resized to 256 pixels and then are center cropped to 224×224 on these four FGVR datasets.

4.2. Evaluation Protocols

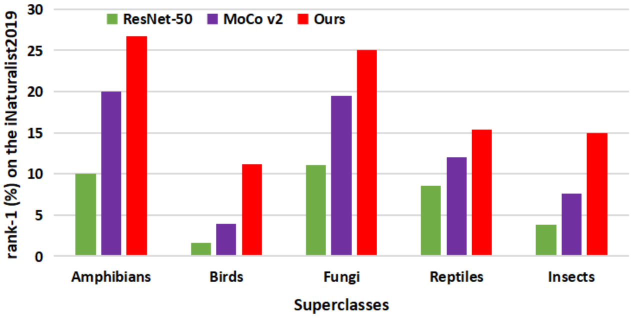

Figure 3. Comparison of ResNet-50, MoCo v2 and Ours on the retrieval rank-1 on the `iNaturalist2019` dataset. “`Amphibians`”, “`Birds`”, “`Fungi`”, “`Reptiles`”, and “`Insects`” are the superclasses in `iNaturalist2019`.

We evaluate the proposed method in two settings: linear probing and image retrieval. Linear probing is a commonly used evaluation protocol in SSL. In linear probing, the feature extractor learned from the SSL algorithm will be fixed and a linear classifier will be trained on top of the learned features. The classification performance of the linear classifier indicates the quality of the learned feature.

Besides linear probing, we also use image retrieval (also equivalent to the nearest neighbor classification task) task to evaluate the learned features, which was also explored in the literature [6, 22, 23]. This task aims to find the images with the same category as query images based on the

Table 2. Classification and retrieval of our method evaluated on the CUB-200-2011, Stanford Cars and FGVC Aircraft datasets. "ResNet-50" represents pre-training in the ImageNet dataset [12] in a supervised manner, then freezing the ResNet-50 backbone and only optimizing the supervised linear classifier in the classification task. We report the Top 1 and Top 5 (in %) on the classification task, rank-1, rank-5, and mAP (in %) on the retrieval task. 100, 50, and 20 are the three different label proportions (in %) in the classification task.

Dataset	Method	Classification			Retrieval		
		Top 1/Top 5(100)	Top 1/Top 5(50)	Top 1/Top 5(20)	rank-1	rank-5	mAP
CUB-200-2011	ResNet-50	68.17/90.42	58.99/85.90	46.54/77.09	10.65	29.32	5.09
	MoCo v2	68.30/90.85	60.96/87.00	46.91/76.59	17.07	41.46	8.13
	Ours	**71.31/92.03**	**66.52/90.06**	**55.33/83.52**	**49.69**	**75.23**	**24.01**
Stanford Cars	ResNet-50	57.41/83.55	46.23/74.31	31.19/58.67	4.91	16.98	2.34
	MoCo v2	58.43/84.85	50.17/77.38	35.14/64.10	10.94	29.57	3.12
	Ours	**60.75/86.44**	**53.87/81.72**	**40.88/69.18**	**34.56**	**60.75**	**8.87**
FGVC Aircraft	ResNet-50	47.38/74.73	37.83/67.12	28.20/54.73	5.16	14.22	2.61
	MoCo v2	52.54/80.74	45.52/73.85	35.17/65.08	19.38	39.90	6.30
	Ours	**55.87/84.73**	**48.22/77.14**	**38.55/68.53**	**34.33**	**61.09**	**15.43**

learned feature. Note that, unlike linear probing, the image retrieval task does not involve a large amount of labeled data – which could be used to suppress the less relevant features. In this sense, succeeding in the image retrieval task imposes higher feature quality requirements. Moreover, image retrieval is a practically useful task, and unsupervised feature learning is an attractive solution to image retrieval since the whole retrieval system can be built without any human annotation and intervention. For the retrieval task, we use rank-1, rank-5, and mAP as evaluation metrics.

4.3. Main Results

The Effectiveness of the Proposed Method. Our method is based on MoCo v2. We first compare our result against MoCo v2 to examine the performance gain. The results are reported in Table 2 and Figure 3.

From Table 2, we can see that our method has led to a significant improvement over MoCo v2. It achieves 71.31% top-1 accuracy and 92.03% top-5 accuracy on the CUB-200-2011 dataset with 100% label proportions, which is a 3.01% improvement on top-1 and 1.18% improvement on top-5 over MoCo v2. Similarly, significant improvement can also be found on the Stanford Cars and FGVC Aircraft datasets. This demonstrates that our methods improve the quality of the feature representations learned from the original MoCo v2. Notably, with the label proportions reduced from 100% to 20%, there is generally a better improvement in the performance of the proposed method, which demonstrates that the proposed method learns less noisy features, so that training with fewer data can generalize better.

The advantage of the proposed method becomes more pronounced when evaluating the image retrieval task. As seen from Table 2, our method leads to a significant boost in performance on all datasets. Specifically, the rank-1, rank-5 and mAP of ours (K=32) are 49.69%, 75.23%, and 24.01%, respectively on the CUB-200-2011 dataset, which is 32.62%, 33.77%, and 15.88% higher than the method of MoCo v2. Similar improvement can also be observed on the Stanford Cars and FGVC Aircraft datasets. On the large-scale fine-grained dataset, i.e., iNaturalist2019, our method also performs better than MoCo v2 shown in Figure 3. These indicate the proposed method is particularly good for retrieval tasks, which might be due to its ability to filter out less relevant patterns.

Comparison with Other SSL Frameworks. In addition to the comparison on MoCo v2, we also compare the proposed method against other commonly used SSL approaches. We report the Top-1 accuracy, the running time, and the peak memory with two different batch sizes (32 & 128). The results are shown in Table 3. All methods are run on 4 V100 GPUs, each with 32G memory. We report the training speed, GPU memory usage, and performance. For classification performance, the blue marks mean the best classification results, and the green marks mean the second-best classification results. It is clear to see that our method achieves the best Top-1 performance on the CUB-200-2011, Stanford Cars, and FGVC Aircraft datasets. Also, when the GPU resources are limited, e.g., only 1 V100 GPU is available, the proposed method reduces the batch size but still remains a competitive classification performance compared to other SSL methods. Regarding training speed and GPU memory usage, the proposed method has the same running time as MoCo v2 and SimSiam, slightly more peak memory than SimSiam, but less peak memory, and quicker running time than SimCLR and BYOL. Although DINO uses the least training time and GPU memory usage, their performances are also the worst compared to ours and other self-supervised learning methods. Barlow Twins and VICReg

Table 3. Compared to other state-of-the-art self-supervised learning frameworks with Top 1 accuracy on the `CUB-200-2011`, `Stanford Cars` and `FGVC Aircraft` datasets; running time and peak memory on the `CUB-200-2011` dataset. The training time is measured on 4 Tesla V100 GPUs with 100 epochs, and the peak memory is calculated on a single GPU. Top 1 accuracy (%) is reported on linear classification with the frozen representations of their feature extractor. For fairness, all following models use the ResNet-50 as the network backbone and initialize the ResNet-50 architecture with ImageNet-trained weights. blue=best, green=second best.

Method	Batch Size	Top 1(`CUB`)	Top 1(`Cars`)	Top 1(`Aircraft`)	time(`CUB`)	GPU Memory(`CUB`)
Supervised	-	81.34	91.02	87.13	-	-
DINO [6]	32/128	12.37/16.66	9.27/10.51	8.52/12.93	1.5h/1.5h	4.9G/8.4G
SimCLR [7]*	32/128	33.49/38.39	44.31/49.41	40.56/45.22	2.5h/2.5h	7.1G/23.8G
BYOL [16]*	32/128	36.64/39.27	43.66/45.21	34.90/37.62	4.0h/4.0h	7.4G/24.6G
SimSiam [9]	32/128	35.82/39.97	56.87/58.89	41.59/43.06	2.0h/2.0h	4.4G/8.9G
MoCo v2 [8]	32/128	68.03/68.30	52.61/58.43	42.51/52.54	2.0h/2.0h	6.1G/10.3G
BarlowTwins [40]	32/128	28.58/33.45	23.34/31.91	28.35/34.77	1.5h/1.5h	7.0G/8.9G
VICReg [4]	32/128	30.07/37.78	19.29/30.80	29.97/36.00	1.0h/1.0h	7.0G/9.0G
Ours	32/128	70.43 /71.31	56.90/60.75	46.92/55.87	2.0h/2.0h	6.1G/10.3G
BYOL+Ours*	32/128	45.79/51.20	48.53 /50.64	40.08/45.94	4.0h/4.0h	7.4G/24.6G

* Due to computational constraints, we are unable to evaluate on the batch size of 4096 as used in the original paper; we leave this for future work.

have a quicker running time and less GPU memory than our proposed method with a batch size of 128, but their performances are much worse than ours.

Our method also can be implemented with other self-supervised learning methods, e.g., BYOL, referring to "BYOL+Ours" presented in Table 3. As we can see, our method applied to the BYOL objective is consistently superior to the baseline of BYOL on three fine-grained datasets.

4.4. Comparison with Alternative Solutions

Our method is featured by its capability of discovering the key discriminative regions, which is vital for FGVR. In this section, we compared the proposed method with nine alternative solutions to take object localization into account. The first is to simply use a bilinear network [26] for MoCo V2. Specifically, we follow the similar bilinear structure as in [33], which implicitly learns K parts and aggregates features from those K parts, and we set $K = 32$ to make a fair comparison to us (since we use 32 projections). We still use MoCo v2 as the SSL framework and denote this method MoCo v2 -Bilinear. The second and third are the methods of *SAM-SSL* and *SAM-SSL-Bilinear* introduced in Section 3.2 since our method is extended from the self-boosting attention mechanism (SAM) proposed in [33]. The fourth and fifth are the two variants considered in Section 3.2. The other four comparing methods are the very recently localization-based self-supervised methods, DiLo [42], CVSA [38], LEWEL [20], and ContrastiveCrop [29]. DiLo uses a copy-and-pasting approach as a kind of augmentation to create an image with a different background. The work of CVSA targets a similar problem as ours. They exploit self-supervised fine-grained contrastive learning via cross-view saliency alignment to crops and swaps saliency regions of images. LEWEL adaptively aggregates spatial information of features using spatial aggregation operation between feature map and alignment map to guide the feature learning better. The work of ContrastiveCrop is based on the idea of using attention to guide image cropping, which localizes the object and improves the data augmentation for self-supervised learning.

The comparison to those nine alternatives is shown in Table 4. As seen, by comparing the *SAM-SSL* and *SAM-SSL-Bilinear*, we observe that the proposed method can lead to overall better performance, achieving a significant boost on some datasets. Occasionally, *SAM-SSL-Bilinear* (K=32) can achieve comparable performance as ours, but at the cost of using a much higher feature dimensionality. This clearly shows the advantage of the proposed scheme over the scheme in [33]. Furthermore, our method and combined with ContrastiveCrop (i.e., ours+ContrastiveCrop) with the lowest feature dimensions but achieve the best Top 1 and rank-1 performance on the `CUB-200-2011`, `Stanford Cars` and `FGVC Aircraft` datasets, compared to those nine alternatives. Also, compared with the other localization-aware SSL methods, our method shows a clear advantage, especially for the retrieval task, e.g., ours vs. LEWEL. Finally, we find that the variant of our method does not produce a good performance. For example, when applying weighted average pooling at both training and testing, i.e., *Ours-DualPooling*, will make the feature learning fail completely, and the representations will collapse. On the other hand, not performing cross-branch interaction, i.e., *Ours-MultiTask*, does not bring too much improvement over the MoCo v2 baseline.

4.5. The Impact of Number of Projections

To explore the impact of the number of linear projections in our method, we conduct experiments with the different

Table 4. The linear Top 1 (%) and retrieval rank-1 (%) performance comparisons of recent alternative solutions and Ours on the `CUB-200-2011`, `Stanford Cars` and `FGVC Aircraft` datasets. blue=best, green=second best. Collapse means the model fails to produce meaningful performance.

Method	Feature	Classification			Retrieval		
	Dimension	`CUB`	`Cars`	`Aircraft`	`CUB`	`Cars`	`Aircraft`
MoCo v2 [8]	C	68.30	58.43	52.54	17.07	10.94	19.38
MoCo v2 -Bilinear	$C*K$	68.44	58.06	53.01	41.27	30.89	30.80
SAM-SSL	C	68.59	58.49	52.97	18.38	14.26	21.72
SAM-SSL-Bilinear	$C*K$	71.56	59.12	55.12	44.20	35.38	32.10
DiLo [38, 42]	C	64.14	-	-	-	-	-
CVSA [38]	C	65.02	-	-	-	-	-
LEWEL [20]	$C*K$	69.27	59.02	54.33	19.23	12.01	20.67
ContrastiveCrop [29]	C	68.82	61.66	54.40	18.71	13.61	20.88
Ours-DualPooling	C	collapse			collapse		
Ours-MultiTask	C	68.56	58.55	52.87	17.62	13.98	22.23
Ours	C	71.31	60.75	55.87	49.69	34.56	34.33
Ours+ContrastiveCrop	C	72.84	63.71	56.08	49.36	33.55	34.95

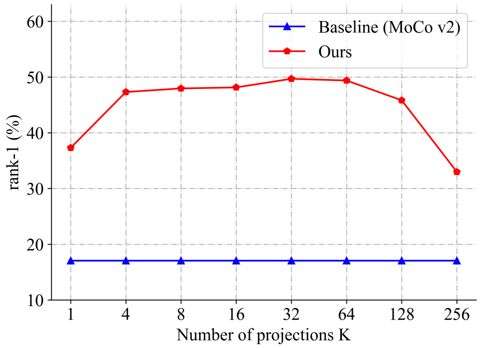

Figure 4. Comparison of MoCo v2 (the blue plot) baseline with our method (the red plot) w.r.t. K on the `CUB-200-2011`.

numbers of K. Figure 4 shows the retrieval results of rank-1 w.r.t. eight different projections on the fine-grained dataset `CUB-200-2011`. As we can see, with the increase of linear projections K, the rank-1 gradually increases—the final rank-1 peaks at 49.69% with K around 32. After K reaches 64, the performance decreases. When the number of projections is very large, the combination of K-part detectors becomes spurious and overfits the correlated pattern, thus resulting in a drop in performance. From the curve, we can also see that choosing any value between 4 and 64 can lead to similar performance. So our method is not very sensitive to K once it falls into a reasonable range.

4.6. Alternative Structure for GFB

In this section, we investigate alternative designs for GFB. In particular, we consider using a two-layer (with 32 as the intermediate feature dimension) multi-layer perception (MLP) to replace the maximized projections in the proposed method. Compared with our GFB structure, an MLP has better fitting capacity due to the extra linear layer. We conduct experiments on the `Stanford Cars` dataset, and the result is shown in Table 5. We find that the performance dramatically decreases when using MLP. This demonstrates the importance of our GFB module design.

Table 5. Retrieval performance (%) of our methods and using MLP as the alternative GFB branch. The evaluation is on the `Stanford Cars` dataset.

Dataset	Architecture	rank-1	rank-5	mAP
`Cars`	Ours	34.56	60.75	8.87
	MLP	25.57	50.91	5.92

5. Conclusion, Further Results (Appendix) , Limitation and Future Work

In this paper, we introduce a simple-but-effective way of learning an additional screening mechanism to self-supervised feature learning for fine-grained visual recognition. The idea is to identify discriminative clues commonly seen across instances and classes by fitting the GradCAM of the SSL loss with a fitting-capability-limited branch. Our method achieves state-of-the-art in the classification task and is particularly pronounced in retrieval tasks on fine-grained visual recognition problems. More experimental results, including adopting our method for non-fine-grained visual recognition problems, and visualizing each projection in the GFB, can be found in the Appendix. So far, the proposed method seems to be most effective for FGVR, which could be a limitation and we plan to extend the applicability of the proposed method in our future work.

References

[1] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Compress: Self-supervised learning by compressing representations. *Advances in Neural Information Processing Systems*, 33:12980–12992, 2020. 3

[2] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*, 32, 2019. 1

[3] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 2

[4] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021. 1, 2, 7

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020. 1, 2

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021. 5, 7

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 2, 3, 7

[8] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 2, 3, 5, 7, 8

[9] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 1, 2, 7

[10] Yuanzheng Ci, Chen Lin, Lei Bai, and Wanli Ouyang. Fast-moco: Boost momentum-based contrastive learning with combinatorial patches. *arXiv preprint arXiv:2207.08220*, 2022. 2, 3

[11] Elijah Cole, Xuan Yang, Kimberly Wilber, Oisin Mac Aodha, and Serge Belongie. When does contrastive visual representation learning work? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14755–14764, 2022. 1

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6

[13] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9588–9597, 2021. 2

[14] Linus Ericsson, Henry Gouk, and Timothy M Hospedales. How well do self-supervised models transfer? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5414–5423, 2021. 1

[15] Yuting Gao, Jia-Xin Zhuang, Shaohui Lin, Hao Cheng, Xing Sun, Ke Li, and Chunhua Shen. Disco: Remedying self-supervised learning on lightweight models with distilled contrastive learning. In *European Conference on Computer Vision*, pages 237–253. Springer, 2022. 2

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020. 1, 2, 7

[17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 2

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 1, 2, 3

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[20] Lang Huang, Shan You, Mingkai Zheng, Fei Wang, Chen Qian, and Toshihiko Yamasaki. Learning where to learn in cross-view self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14451–14460, 2022. 7, 8

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015. 5

[22] Young Kyun Jang and Nam Ik Cho. Self-supervised product quantization for deep unsupervised image retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12085–12094, 2021. 5

[23] Donghyun Kim, Kuniaki Saito, Tae-Hyun Oh, Bryan A Plummer, Stan Sclaroff, and Kate Saenko. Cds: Cross-domain self-supervised pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9123–9132, 2021. 5

[24] Nikos Komodakis and Spyros Gidaris. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*, 2018. 2

[25] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013. 5

[26] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1457, 2015. 7

[27] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 5

[28] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 2

[29] Xiangyu Peng, Kai Wang, Zheng Zhu, Mang Wang, and Yang You. Crafting better contrastive views for siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16031–16040, 2022. 2, 7, 8

[30] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 1, 2, 3

[31] Ramprasaath R Selvaraju, Karan Desai, Justin Johnson, and Nikhil Naik. Casting your model: Learning to localize improves self-supervised representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11058–11067, 2021. 2

[32] Yuge Shi, N Siddharth, Philip Torr, and Adam R Kosiorek. Adversarial masking for self-supervised learning. In *International Conference on Machine Learning*, pages 20026–20040. PMLR, 2022. 2

[33] Yangyang Shu, Baosheng Yu, Haiming Xu, and Lingqiao Liu. Improving fine-grained visual recognition in low data regimes via self-boosting attention mechanism. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pages 449–465. Springer, 2022. 4, 5, 7

[34] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018. 5

[35] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 5

[36] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020. 2

[37] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3024–3033, 2021. 2

[38] Di Wu, Siyuan Li, Zelin Zang, Kai Wang, Lei Shang, Baigui Sun, Hao Li, and Stan Z Li. Align yourself: Self-supervised pre-training for fine-grained recognition via saliency alignment. *arXiv preprint arXiv:2106.15788*, 2021. 2, 7, 8

[39] Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, and Tyng-Luh Liu. Saga: Self-augmentation with guided attention for representation learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3463–3467. IEEE, 2022. 2

[40] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021. 7

[41] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016. 2

[42] Nanxuan Zhao, Zhirong Wu, Rynson WH Lau, and Stephen Lin. Distilling localization for self-supervised representation learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10990–10998, 2021. 2, 7, 8

[43] Mingkai Zheng, Shan You, Fei Wang, Chen Qian, Changshui Zhang, Xiaogang Wang, and Chang Xu. Ressl: Relational self-supervised learning with weak augmentation. *Advances in Neural Information Processing Systems*, 34:2543–2555, 2021. 2