CAM Back Again: Large Kernel CNNs from a Weakly Supervised Object Localization Perspective

Supplementary Material

8. Training Configurations

In this study, models pre-trained on the ImageNet1K dataset are fine-tuned for the CUB-200-2011 dataset. 12 fine-tuned models are trained from 6 pre-trained models in RepLKNet, 14 fine-tuned models are trained from 7 pre-trained models in ConvNeXt, and 10 fine-tuned models are trained from 5 pre-trained models in SLaK. As a preprocessing step for the CUB-200-2011 dataset for training, we resize images to square size of 512 x 512 and then apply center cropping at a ratio of 0.875 to obtain 448 x 448 images.

We first present the details of RepLKNet and ConvNeXt. For RepLKNet fine-tuning, we use the following 6 pre-trained models provided by [13]; 31B1K224, 31B1K384, 31B22K224, 31B22K384, 31L22K384, XL. For ConvNeXt fine-tuning, we use the following 7 pre-trained models provided by PyTorch Image Models (timm) library; convnext-tiny, convnext-small, convnext-base, convnext-base-in22ft1k, convnext-base-384-in22ft1k, convnext-large, convnext-large-in22ft1k. With 8 GPUs, NVIDIA Quadro RTX 8000, we train 400 epochs with a batch size of 16 per GPU. The learning rate is 10^{-4} for the FC layer and 10^{-5} for the rest. The input resolution is set to the resolution of the pre-trained model, and other settings follow the fine-tuning settings of [13]. With or without data augmentation in fine tuning, two fine-tuned models are obtained from one pre-trained model.

Next, we present the details of SLaK. The pre-trained model uses SLaK-T (K=31×31), SLaK-T (K=51×51), SLaK-T (K=61×61), SLaK-S (K=51×51), SLaK-B (K=51×51) provided by [23] (K is kernel size). They are all trained at 224 x 224 resolution, but the difference in whether the fine-tuning pixel size setting is 224 x 224 or 384 x 384 results in two fine-tuned models from a single pre-trained model. With 8 GPUs, NVIDIA Quadro RTX 8000, we train 400 epochs with a batch size of 16 per GPU. The drop-path is set to 0.1, 0.4, and 0.5 for tiny, small, and base, respectively, and update-frequency is set to 2000, 10, and 10, according to the recommendations of [23]. Other settings follow the fine-tuning settings of [23]. These models follow the training protocol for classification and are not optimized for WSOL (e.g., we do not evaluate WSOL scores at each epoch). And our high-score, the 100-epoch RepLKNet31B1K384 model's 90.99%MaxBoxAcc in the CUB-200-2011 dataset, actually reports the best score of the checkpoint models saved every 50 epochs.

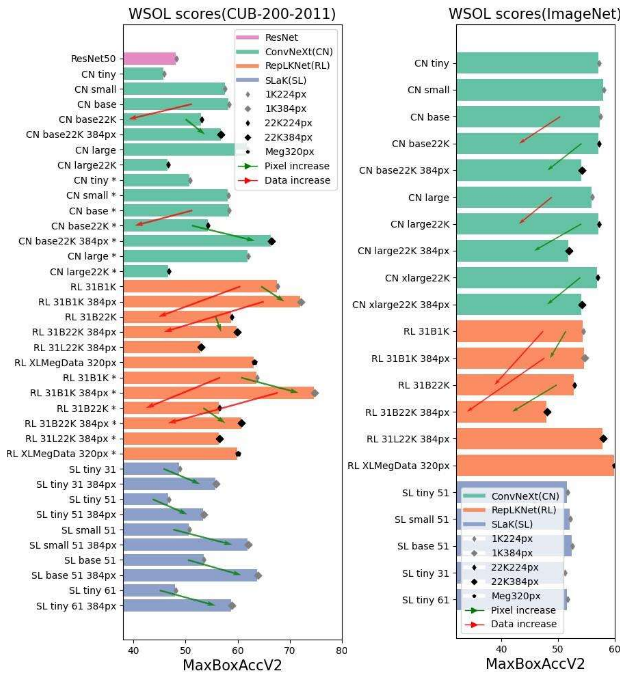

Figure 13. WSOL scores (MaxBoxAccV2) for the CUB-200-2011 dataset on the left and the ImageNet1K dataset on the right. Green arrows connect models that differ only in training pixel size settings, and red arrows connect models that differ only in pre-training settings. * denotes models that introduced the same data augmentation strategy as the pre-training in fine tuning.

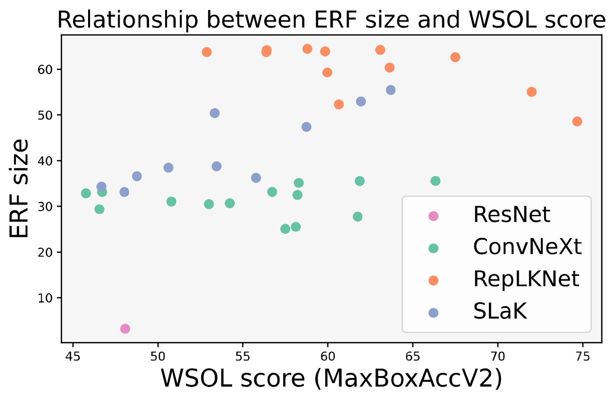

Figure 14. Graph shows the relationship between WSOL score (MaxBoxAcc2) and ERF size. Here, ERF size is measured by the Area Under the Curve (AUC) of Fig. 10 (a).

9. WSOL Evaluation Details

Our WSOL evaluation methodology is largely based on the [9]. For the CUB-200-2011 dataset, we use 5994 images (train-weaksup) for training (fine-tuning) and 5774 images (test) for testing, as well as 1000 images (train-fullsup) for threshold search. There is a problem with the train-fullsup label information provided in the ImageNet1K (ILSVRC2012) dataset [2]. Therefore, threshold search and testing will be performed by a test set of 50,000 images. Apart from MaxBoxAcc, which is primarily reported in this paper, here we present the results of MaxBoxAccV2 (Figs. 13 and 14). Checking the direction of the arrows, we can see the same trend as in the MaxBoxAcc graph Fig. 13.

10. Analysis on ERF and Shape Bias

The latest research [34] argues that Vision Transformers are more human-like in that they tend to make predictions based on the overall shape of objects, whereas CNN is more likely to be based on local texture. In contrast, [13] show that RepLKNet has a higher shape bias than Swin Transformer and small kernel CNNs. Furthermore, from the perspective of comparison with the Swin Transformer, they argue for the possibility that the strength of the shape bias is closely related to ERFs. Therefore, we performed similar experiments on untested RepLKNet and other latest CNNs by [13] using the toolbox [1] (Fig. 16). Fig. 10 (a) shows that ERF is larger for ConvNeXt, SLaK, and RepLKNet, but Fig. 16 shows that the strength of shape bias is stronger for SLaK, ConvNeXt, and RepLKNet, in that order. These facts negate any simple correlation between the strength of the shape bias and ERF. Also, considering our findings in this paper, the strength of the shape bias seems to be related to the model's capability to generate feature maps with large activation regions.

On the other hand, from a more microscopic perspective, the relationship between shape bias and ERFs can be seen. As an example, consider the clock category in Fig. 16. Clock is a category with strong shape bias in many models—an image whose shape is a clock and whose style is another category (e.g., elephant) is likely to be predicted as a clock category. Among the images whose shape is a clock, the ERFs obtained from the group of images predicted to be in the clock category and predicted to be in the style side category show that the former ERF is larger for most models (Fig. 17). These results can provide clues to understanding the perceived usefulness of large ERFs, including long-range dependencies.

11. CAM(Class Activation Map) Calculation Method and Semantic Interpretation

Class Activation Map (CAM) is a method for localizing object regions in an image that belong to a specific category using CNN with Global Average Pooling (GAP). It is a typical method in WSOL tasks and is often used as a baseline method. To calculate CAM, the feature maps of the final convolutional layer and the weights of the FC layer that connect the feature vectors obtained by GAP of those maps to the final logit are used (Fig. 15). The feature maps of the final convolutional layer is F, the n-th channel slice is $F_n \in \mathbb{R}^{I\times J}$, and the size of the feature map F_n is $I \times J$. The feature map F in original CNN architecture is transformed into a feature vector $G \in \mathbb{R}^N$ by GAP layer. The scalar G_n corresponding to the feature map F_n is the glocal average pooling of F_n

$$G_n = \frac{1}{I \times J}\sum_i \sum_j F_n(i,j). \tag{1}$$

The feature vector G obtained here is transformed into a C-dimensional logit vector through the final FC layer (C is the number of classes). Let W be the weight of the FC layer of size $N \times C$. The weight between the feature scalar G_n and the final logit for the class c is $W_{n,c} \in \mathbb{R}$. The class activation map $M_c \in \mathbb{R}^{I\times J}$ for class c is then given by the following weighted sum over the channel dimension

$$M_c = \sum_{n=1}^{N} W_{n,c} \cdot F_n. \tag{2}$$

The semantic interpretation of the CAM is as follows. In the classification model, the N weights between the N elements of feature vector $G \in \mathbb{R}^N$ and the final logit for a given class c can be interpreted as carrying information about how important each feature element $G_n \in \mathbb{R}$ should be to obtain correct class predictions. In addition, spatially meaningful information is stored in the feature map of the CNN. Therefore, by assigning this weight as the importance of each feature map $F_n \in \mathbb{R}^{I\times J}$, spatially meaningful regions emerge from the feature map set.

In the WSOL task, binarization by threshold is used to obtain the prediction results of localization from the generated heatmap as a bounding box. According to the [9], the determination of the threshold value should be calculated using a different dataset (fullsup dataset) than the data for classification model training or the final prediction. The method of obtaining the final predicted regions from the binarized map depends on the evaluation method (MaxBoxAcc or MaxBoxAccV2).

12. Details of Problems Faced by CAM in WSOL Tasks

This chapter provides a detailed description of the problems CAM faces in the WSOL task shown in Fig. 2. The two problems shown in the figure are also discussed in [3].

First, for the three feature maps and weights indicated by F_i in the upper row in Fig. 2. In training for classification,

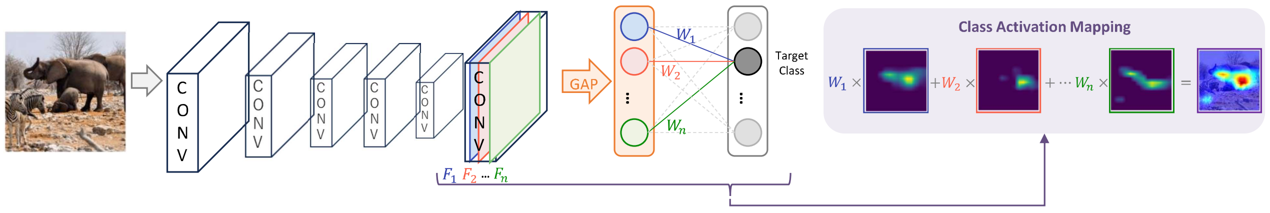

Figure 15. Calculation process of CAM. To calculate CAM, the feature maps of the final convolution layer and the weights of the FC layer, which connect the feature vectors obtained by global average pooling of those maps to the final logit, are used.

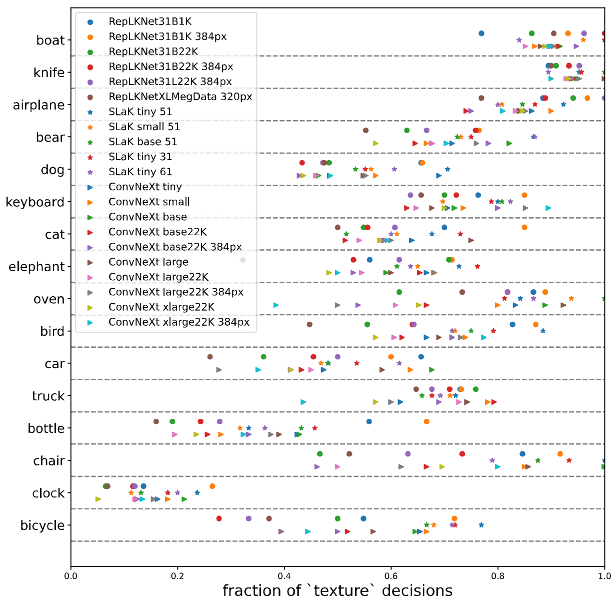

Figure 16. Texture bias and shape bias measurements in the latest CNNs.

the smaller (larger) the activation area of a feature map, the larger (smaller) the weight tends to be, and F_i in the figure represents such a state. If the CAM is calculated under such conditions, it will look like the CAM in the upper right corner. The central feature map, which has a small activation area but large weights, has a significant contribution to the generated CAM, resulting in a CAM in which the bird's head is particularly activated. For clarity, the CAM is generated here from only three feature maps corresponding to positive weights (F_{pos}), but this is a known problem that occurs regardless of the plus/minus of the weights or the number of maps.

Next, for the three feature maps and weights indicated by F_j in the bottom row. These feature maps activate portions of the object region that are not important for classification. The activation of feature maps corresponding to negative weights (hereafter referred to as F_{neg}) should ideally not be activated in the classification task, since it only reduces

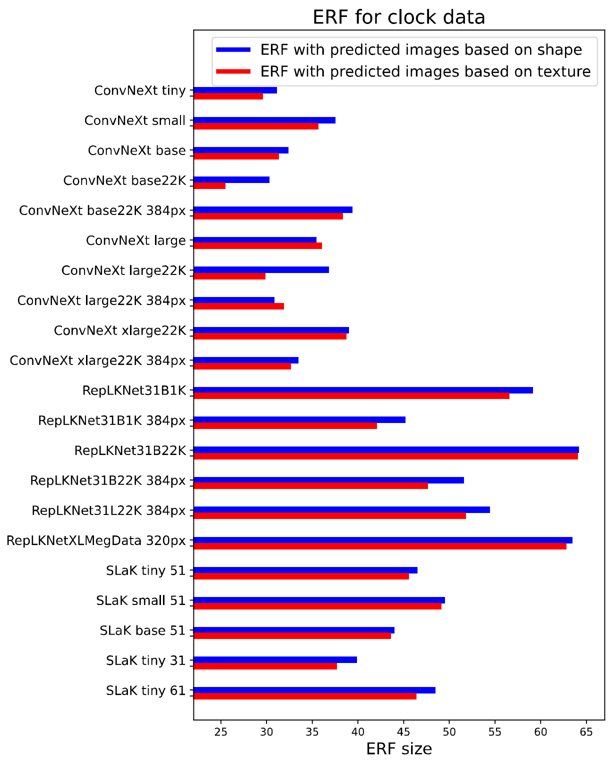

Figure 17. Graph comparing ERF sizes obtained from different image groups. For images whose shape is clock and whose style is the other category, the ERF sizes obtained from the group of images predicted to be in the clock category are shown in blue. The ERF sizes obtained from the group of images predicted to be in the style side category are shown in red. The calculation of ERF size is the same as in Fig. 10 (a).

the activation value of the final logit for the correct target class. However, F_{neg} are generally also activated. In this case, the activated region should be at least a no-object region to avoid adversely affecting the shape of the CAM. It is however often activated within an object region, such as F_j. F_{neg} works to deactivate those activated regions when computing the CAM. In this example, the areas of the bird's abdomen and feet are deactivated. Thus, when the CAM for

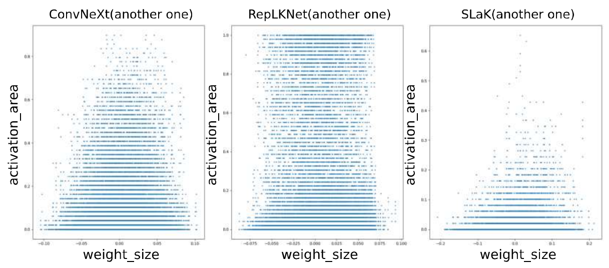

Figure 18. Relationship between activation areas of the feature maps and weights. The activation area is calculated by binarizing the feature map with a threshold of 10 and calculating the percentage of pixels that exceed the threshold. Experiments on ConvNeXt, RepLKNet, and SLaK, fine-tuned on the CUB-200-2011 dataset.

F_i, which was already locally activated, is combined with the CAM for F_j, it becomes an even more locally activated CAM (bottom right CAM).

13. More Analysis of Activation Area Size and Weight Size

Each plot in Fig. 18 shows the relationship between activation areas of the feature maps and weights. This is the same experiment as in Fig. 5, but using a model that did not have the best WSOL score. The model identifiers used are `ConvNeXt_tiny`, `RepLKNet31B22K224`, and `SLaK_tiny51_224`. The model identifiers with the best WSOL scores (Fig. 5) are `ConvNeXt_base_384_in22ft1k`, `RepLKNet31B1K384`, and `SLaK_base51_384`.

Fig. 19 shows the relationship between GAP values of the feature maps and weights. This is the same experiment as in Fig. 12, but using a model that did not have the best WSOL score. The non-best scoring model used are the same as in Fig. 18.

Each figure shows that similar trends discussed in the main text can be observed even when the model is changed.

14. Quantification of Feature Maps Complexity

For the reasons discussed in Sec. 4.5, we believe that Fig. 8 is material to quantify the complexity of the feature maps. In addition, for more direct quantification, we performed an analysis using dictionary learning. Specifically, the dictionary is learned from the sampled feature maps using Orthogonal Matching Pursuit method, and feature maps not used for training are reconstructed using the dictionary. The more complex the feature map set, the larger the reconstruction error is expected to be. Fig. 20 shows the reconstruction errors when the number of components in the dictionary is varied. Regardless of the number of components, the relationship between the three models is constant, with ConvNeXt having the largest error. In addition, the relationship between the three models and the results in Fig. 8 is consistent and supports our assertion. Note that SLaK feature maps of low complexity are exceptional in quality and size (Fig. 3, Fig. 7, Tab. 3).

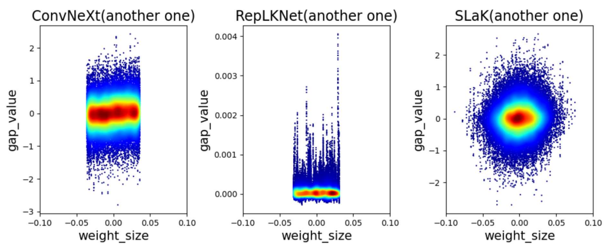

Figure 19. Relationship between GAP values of the feature maps and weights. Experiments on ConvNeXt, RepLKNet, and SLaK. Models with non-best WSOL scores in CUB-200-2011 dataset are used and the weights are randomly initialized. Plotted against 100 groups of feature maps obtained from 100 random images selected from the same dataset.

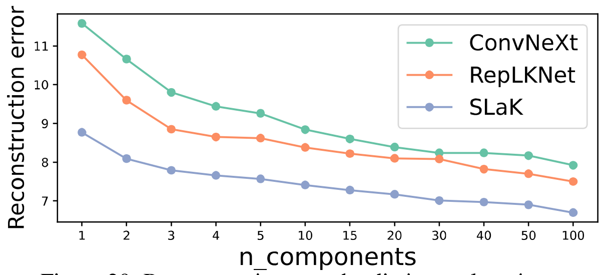

Figure 20. Reconstruction error by dictionary learning.

15. What Does the Architecture Need to Achieve the Desired Characteristics?

What is needed in the architecture to get a feature maps like ConvNeXt or RepLKNet? Based on our results so far, we believe it is important to use a simple, large kernel that does not involve decomposition into rectangular kernels or spercity as used in SLaK. This is an insight gained from the fact that SLaK inherits most of its architecture from ConvNeXt, yet does not have the desired characteristics, and the similarities between ConvNeXt and RepLKNet, which have very different architectures. However, answering this question accurately requires a more comprehensive examination and is an interesting future study.

16. Additional material on CAM quality differences between RepLKNet and ConvNeXt

As mentioned in Sec. 4.4, PC1 in ConvNeXt and RepLKNet seem to be well suited for localization. Therefore, we actually generated a localization map based on the binarized PC1 heatmap and used it to measure WSOL scores (Tab. 3). As a result, RepLKNet scored higher than ConvNeXt. Furthermore, ConvNeXt's IoU threshold [9] is very high (Tab. 3). The high IoU threshold implicitly indicates the large activation area. For example, if the activation area tends to be larger than the actual object area, the threshold should be higher to round unwanted activation to zero for localization. In fact, when several WSOL methods are applied to ResNet, the optimal IoU threshold is between 10 and 40 [9]. Additionally, the average optimal thresholds for the CAMs of ConvNeXt, RepLKNet, and SLaK in this paper were 45, 28.5, and 4.1, respectively. Thus, the threshold for ConvNeXt+PC1 is high and can be interpreted as large PC1 activation. This result is another material that the quality of RepLKNet's PC1 is superior to that of ConvNeXt.

Interestingly, the WSOL score based on PC1 is even higher than the CAM score discussed in this paper (Tab. 1, Fig. 1), with a RepLKNet MaxBoxAcc of 93.6% (Tab. 3). To our knowledge, this is 0.43% higher than the state-of-the-art performance of CNN-based WSOL (Tab. 1). Simply put, this new WSOL method uses PC1 instead of feature maps in CAM. See Sec. 17 for details on the methodology.

17. Details of WSOL Method Using PC1

This section provides details on the WSOL methodology reported in Sec. 4.5 and Sec. 16. In this method, localization maps are generated from PC1 features obtained by PCA. As shown in Fig. 7, the visualized PC1 features do not tell whether the object region is white or black. This method contains an ingenious solution to this problem. The procedure is as follows. (1) Perform PCA instead of GAP in CAM to obtain PC1 features. (2) Binarize PC1 using the average value of all pixels in PC1 as the threshold value. (3) Calculate the average value of the outer edge pixels of the binary map (e.g., 44 pixels for a 12x12 map). (4) For example, if the average value (3) is close to 0, the object area shall be indicated by 1. This method has some characteristics that should be noted, including the lack of class discriminability and the fact that it is heavily influenced by the quality of the feature maps. However, high WSOL performance can be expected for models with high-quality feature maps. The code for our experiments is available at: https://github.com/snskysk/CAM-Back-Again.