Multi-view Feature Augmentation with Adaptive Class Activation Mapping

Xiang Gao[1,2,3], **Yingjie Tian**[2,3*] and **Zhiquan Qi**[2,3]

[1]School of Computer Science and Technology, University of Chinese Academy of Sciences
[2]Research Center on Fictitious Economy and Data Science, Chinese Academy of Sciences
[3]Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences
gaoxiang181@mails.ucas.ac.cn, tyj@ucas.ac.cn, qizhiquan@foxmail.com

Abstract

We propose an end-to-end-trainable feature augmentation module built for image classification that extracts and exploits multi-view local features to boost model performance. Different from using global average pooling (GAP) to extract vectorized features from only the global view, we propose to sample and ensemble diverse multi-view local features to improve model robustness. To sample class-representative local features, we incorporate a simple auxiliary classifier head (comprising only one 1×1 convolutional layer) which efficiently and adaptively attends to class-discriminative local regions of feature maps via our proposed AdaCAM (Adaptive Class Activation Mapping). Extensive experiments demonstrate consistent and noticeable performance gains achieved by our multi-view feature augmentation module.

1 Introduction

Progresses of image classification were dramatically promoted by deep convolutional neural networks. NIN [Lin *et al.*, 2013] for the first time replaced traditional flattening-based fully-connected (FC) layers with global average pooling (GAP), which reduces model size and enables input images of arbitrary size. Since then, GAP is widely used in combination with more complex convolutional backbones. GoogLeNet [Szegedy *et al.*, 2015] extracts and fuses multi-scale convolutional features to enrich feature representation. ResNet [He *et al.*, 2016] introduces residual learning to facilitate optimization of deep networks, improving model performance noticeably. Later on, more variants and derivatives of ResNet are proposed to make further improvements by increasing network width [Zagoruyko and Komodakis, 2016], adopting multi-path group convolution [Xie *et al.*, 2017], extending skip connection to dense connections [Huang *et al.*, 2017], and introducing attention mechanism [Wang *et al.*, 2017; Zhang *et al.*, 2020]. Besides, networks were also specially designed for lightweight deployment by using more efficient convolution or convolutional blocks [Howard *et al.*, 2017; Iandola *et al.*, 2016]. However, all these advanced deep models only focus on delicate design of convolutional backbone networks, the common ground is applying GAP to obtain vectorized features which are then fed to a linear classifier head for final classification. Though effective and efficient, GAP extracts vectorized image representations simply by averaging cross the entire feature maps. In this way, the extracted image representations could be corrupted by background elements for images where the class-related object takes up only a fraction of the image. As Figure 1 displays, even for images of the same class, the scale of class-related object varies a lot in ImageNet dataset. For images with a small scale of class-related object, GAP easily extracts image representations that bias towards features of the background or class-irrelevant objects, making the input features to the subsequent linear classifier head less class-representative.

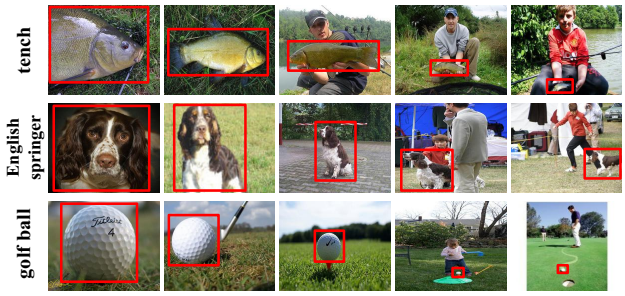

Figure 1: Example images of "tench", "English springer", "golf ball" categories in ImageNet dataset. Even for images of the same class, the class-related objects, which we annotate with red boxes, vary a lot in scale. We consider that for an image with a small scale of class-related object, the final image representation extracted by global average pooling (GAP) could be corrupted by class-irrelevant background features, and thus is less representative of the corresponding class. This motivates us to attend to local region of class-related object and extract more class-representative image representations by sampling local features around the attended region.

Based on this motivation, we replace GAP with a multi-view feature augmentation module (MV-FeaAug), which learns to locate region of class-related object on the final convolutional feature maps, and extracts multi-view image representations by sampling diverse local features around the attended class-discriminative region. The extracted multi-view local representations more precisely represent the class-related object of the image from diverse views, the ensemble

*Contact Author

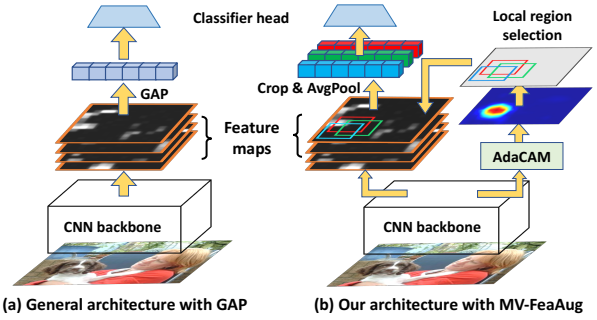

Figure 2: Comparison between the image classification architecture with the general GAP (left) and our MV-FeaAug module (right). We sample diverse local features around the class-discriminative region of the final convolutional feature maps as multi-view local image representations for ensembled classification, as compared with GAP that extracts only global-view image representation.

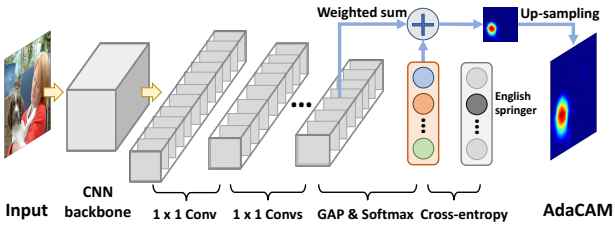

Figure 3: Adaptive class activation mapping (AdaCAM). We replace the traditional classifier head made up of [GAP→MLP→Softmax] with [MLPConv→GAP→Softmax] (MLPConv comprises consecutive $\text{Conv}_{1\times1}$ layers joined by non-linear activations) to maintain spatial resolution of feature maps. The AdaCAM is obtained by performing channel-wise weighted sum of the last convolutional feature maps with respect to the softmax logit vector.

of the predictions to all these representations are used as the final prediction of the entire image. Comparison between the image classification architecture with the general GAP and our MV-FeaAug counterpart is illustrated in Figure 2.

Apart from network architecture design, data augmentation, which acts as a regularizer to reduce overfitting, is also important for image classification. In addition to the basic position and color transformations, some advanced methods were proposed in recent years. Mixup [Zhang *et al.*, 2017] blends two images and their corresponding labels with a random proportion as training samples. Cutout [DeVries and Taylor, 2017] randomly removes a rectangular patch from images to guide the network to learn from local views. CutMix [Yun *et al.*, 2019] randomly cuts and pastes rectangular patches among training images with their corresponding labels mixed up in proportion to patch area. Our method shares similar spirit in expanding training samples with these methods. But instead of image-level augmentation, our method extracts diverse multi-view local representations at feature space as augmented samples for subsequent classifier head, for which we call feature augmentation. Besides, our method takes full advantage of the augmented features at inference time by ensembling predictions to all the multi-view features, while such ensemble prediction property is not available for the above-mentioned data augmentation methods.

Our method also relates to "visual explanation" of CNN models. Efforts have been made to visualize the evidence of class predictions made by CNN. CAM [Zhou *et al.*, 2016] produces class activation maps that highlight class-discriminative regions of images with GAP and a single-layer classifier head. Grad-CAM [Selvaraju *et al.*, 2017] extends such object localization ability to arbitrary CNN architectures by computing gradients of the logits with respect to intermediate feature maps. However, both CAM and Grad-CAM operate in a way of postprocessing since they require extraction (or computation) of weights (or gradients) associated to the target-class logit. Moreover, the attention maps produced by these two methods are conditioned on the ground truth label, which is not flexible in cases when the target image label is unknown. In this work, we propose AdaCAM that adaptively generates attention maps with simply a forward pass of the network without conditioning on the class label. The produced attention maps provide cues to sample diverse class-representative local features for feature augmentation.

Another topic related to our method is attention mechanism, which has been widely applied to image classification recently. SENet [Hu *et al.*, 2018] built a Squeeze-and-Excitation module to selectively enhance or depress features in channel dimension. CBAM [Woo *et al.*, 2018] sequentially stacked two modules for channel-wise and spatial attentions respectively. SKNet [Li *et al.*, 2019] proposed a multi-branch attention module for adaptive convolution kernel size selection. GCNet [Cao *et al.*, 2019] improved model performance by designing an efficient self-attention block that captures global context information. All these methods increase model size for incorporating additional attention modules. By contrast, our method also introduces attention mechanism by locating class-discriminative regions via AdaCAM, however, all the extra cost is an auxiliary classifier head comprised of only one 1×1 convolutional ($\text{Conv}_{1\times1}$) layer. Furthermore, we demonstrate that a classification network with a deeper CNN backbone could be surpassed by a network with a shallower backbone combined with our MV-FeaAug module.

2 Method

In this section, we firstly describe our method for efficiently producing label-independent attention maps, i.e., *adaptive class activation mapping* (AdaCAM), and then elaborate on the complete architecture of our MV-FeaAug module.

2.1 Adaptive Class Activation Mapping

We start with a brief introduction of CAM and Grad-CAM for completeness and ease of comparison.

CAM uses GAP and a single-FC-layer classifier head to produce class activation maps. Supposing the output feature maps of the CNN backbone are $\{F_k\}_{k=1}^K$, where F_k is the k-th channel, and $\{u_k\}_{k=1}^K$ are the units after GAP, i.e., $u_k = \frac{1}{HW}\sum_{x,y} F_k(x,y)$, where H and W denote the height and width of feature maps. The class activation map for class c is $A_c = \sum_k w_k^c F_k$, where w_k^c is the weight of the subsequent FC layer that corresponds to class c for unit u_k. The

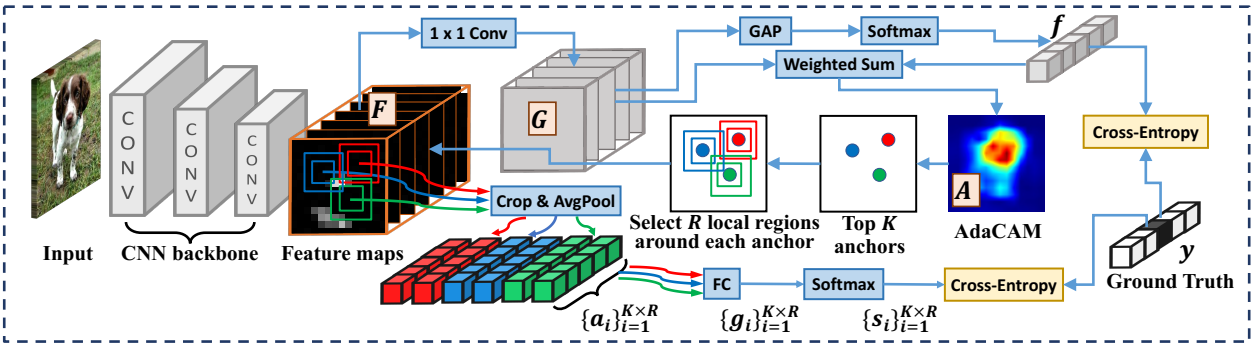

Figure 4: Overview of MV-FeaAug. We concurrently train an auxiliary classifier head comprised of only one 1×1 convolutional layer for dynamic generation of AdaCAM, based on which we sample multiple local representations on the final convolutional feature maps as multi-view inputs to the main classifier head. The main classifier head comprises (but not restricted to) a single fully-connected layer.

limitation of CAM is threefold: (i) it is less efficient for the need of extracting intermediate weights of the model; (ii) it is limited to single-layer classifier head, and thus may compromise classification performance as compared with MLP classifier head; (iii) it is conditioned on the ground truth label, which may not be known in practical applications.

Grad-CAM addresses the second limitation of CAM by backpropagating gradients with respect to convolutional feature maps. Specifically, the class activation maps for class c is $A_c = \sum_k \alpha_k^c F_k$, in which α_k^c is the gradient of f^c, the logit (before softmax) of class c, with respect to F_k, i.e., $\alpha_k^c = \frac{1}{HW}\sum_{x,y} \partial f^c / \partial F_k(x,y)$. However, Grad-CAM is still not efficient for the need of gradient computation, and also relies on the specific class label.

A key ingredient of our MV-FeaAug module is to employ class activation maps to highlight class-related regions for local features sampling, for which a good method to produce such attention maps should follow two criteria: (i) the generation process is simple and efficient; (ii) the generation process is independent of class label, since the ground truth labels are not given at inference time. Both CAM and Grad-CAM fail to satisfy these two conditions, for which we propose AdaCAM to meet the requirements.

AdaCAM aims to directly synthesize class activation maps in a feed-forward pass without conditioning on target labels. This functionality is achieved with only a slight modification of the classifier head. As illustrated in Figure 2, after feature extraction of CNN backbone, we alter the traditional classifier head structure of [GAP→MLP→ Softmax] into [MLPConv→GAP→Softmax], where MLPConv refers to consecutive $\text{Conv}_{1\times1}$ layers joined by non-linear activations. The $\text{Conv}_{1\times1}$ layers are essentially equivalent to FC layers, but have the advantage of maintaining spatial resolution of feature maps. The output of MLPConv is feature maps $\{G_c\}_{c=1}^C$ with its channel number equalling the number of classes C, i.e., $\{G_c\}_{c=1}^C$=MLPConv($\{F_k\}_{k=1}^K$), where G_c represents the c-th channel. Then GAP and softmax operations are employed to obtain the final softmax logit vector $\boldsymbol{f}$, i.e., $\boldsymbol{f}$=Softmax(GAP($\{G_c\}_{c=1}^C$)), the objective is to minimize the cross-entropy between $\boldsymbol{f}$ and the one-hot label vector $\boldsymbol{y}$. This architecture is equivalent to traditional MLP classifier head since MLPConv shares the same parameter space with MLP. The advantage is the efficiency in obtaining class activation maps: for an image of class c', the corresponding class activation map $A_{c'}$ is exactly $G_{c'}$, the c'-th channel of the MLPConv output (we prove this in supplementary materials). To further make the attention map independent of class label, we perform weighted sum on $\{G_c\}_{c=1}^C$ with respect to the softmax logit vector $\boldsymbol{f}$ as an approximation, i.e., $A = \sum_{i=1}^C G_i f_i$, where f_i is the i-th element of $\boldsymbol{f}$. Actually, A is very close to $A_{c'}$ since softmax operation exponentially widens the gap between elements of the logit vector.

In this work, we build a separate auxiliary classifier head tailored for AdaCAM to provide attention maps for the main classifier head. Since the auxiliary classifier is not necessarily to be sufficiently accurate, we use only one $\text{Conv}_{1\times1}$ layer in the MLPConv part to reduce additional overhead.

2.2 Multi-view Feature Augmentation

The overall architecture of MV-FeaAug is illustrated in Figure 4. An auxiliary classifier head is built on top of $\boldsymbol{F} = \{F_k\}_{k=1}^K$, the final feature maps of CNN backbone, to dynamically generate AdaCAM attention maps, based on which we sample multi-view local representations from $\boldsymbol{F}$ as input to the main classifier head. The final prediction at inference time is the ensemble of the predictions made by the main classifier head to all the sampled multi-view representations.

In the auxiliary classifier part, we first use a MLPConv network comprised of a single $\text{Conv}_{1\times1}$ layer to reduce the depth of feature maps to the class number C, i.e., $\boldsymbol{G} = \{G_c\}_{c=1}^C$=$\text{Conv}_{1\times1}(\boldsymbol{F})$, then the softmax logit vector $\boldsymbol{f}$ is obtained after GAP and softmax operations, i.e., $\boldsymbol{f}$=Softmax(GAP($\boldsymbol{G}$)), the objective function is the cross-entropy between $\boldsymbol{f}$ and the one-hot label vector $\boldsymbol{y}$:

$$L_{global} = -\sum_{i=1}^{C} y_i log(f_i), \tag{1}$$

where y_i and f_i are the i-th element of $\boldsymbol{y}$ and $\boldsymbol{f}$ respectively. We call this global loss as our auxiliary classifier considers only the global view of image features by using GAP. The minimization of the global loss endows the auxiliary classifier head object localization ability, which allows for dynamic generation of AdaCAM attention map A:

$$A = \sum_{i=1}^{C} G_i f_i. \tag{2}$$

The attention map A shares the same spatial size as feature maps $\boldsymbol{F}$, and highlights class-discriminative local regions of the image. The pixel intensity of A reflects the correlation of each location with the corresponding image class. Therefore, we select top K locations with the largest pixel values from A as anchor points, around which we sample multiple local regions from $\boldsymbol{F}$ for feature augmentation. Let V be the set of anchor points for an input image, S be an ordered list of the pixel coordinates of A:

$$\begin{aligned} &S = \{(x_i, y_i) | i = 1, 2, ..., HW, \\ &A(x_1, y_1) \geq A(x_2, y_2) \geq ... \geq A(x_{HW}, y_{HW})\}, \end{aligned} \tag{3}$$

where H and W denote the height and width of A, then the anchor set V is the top K subset of S:

$$V = \{(x_i, y_i) | (x_i, y_i) \in S, i = 1, 2, ..., K\}. \tag{4}$$

The number of anchor points K for each input image is a hyper-parameter of the model. Each anchor point in V serves as a centroid around which we crop R multi-scale square regions on $\boldsymbol{F}$ for feature augmentation. For feature maps $\boldsymbol{F}$ with spatial size $H \times W$, we pre-define a region size list $L_R = [r_1, r_2, ..., r_R]$, in which r_i (odd number) denotes the length of the side of the i-th square region to be cropped around each anchor point in V. Specifically, given a centroid $(x_c, y_c) \in V$ and a region length $r \in L_R$, the square region to be cropped on $\boldsymbol{F}$ is determined by the coordinates of the top-left corner (x_{tl}, y_{tl}) and the bottom-right corner (x_{br}, y_{br}):

$$\begin{cases} x_{tl} = max(x_c - (r-1)/2, 0) \\ y_{tl} = max(y_c - (r-1)/2, 0) \\ x_{br} = min(x_c + (r-1)/2, W-1) \\ y_{br} = min(y_c + (r-1)/2, H-1). \end{cases} \tag{5}$$

The length and the elements of L_R are also hyper-parameters of our method. The purpose of sampling multi-scale local regions is to (i) capture class-related objects with different and varying spatial sizes, (ii) further increase the number of the augmented features to reduce overfitting, (iii) extract features of the class-related objects from more views to improve classification robustness. In this way, there are $K \times R$ local feature maps cropped out from $\boldsymbol{F}$ for each input image, to which average pooling is applied separately, yielding the augmented vectorized features $\{\boldsymbol{a}_i\}_{i=1}^{K \times R}$ as multi-view inputs to the main classifier head, as compared with the traditional single-view input GAP($\boldsymbol{F}$). Like most classic networks [He *et al.*, 2016; Szegedy *et al.*, 2015; Xie *et al.*, 2017; Howard *et al.*, 2017], the main classifier head comprises a single FC layer that converts the vectorized features into logit vectors $\{\boldsymbol{g}_i\}_{i=1}^{K \times R}$, i.e., $\{\boldsymbol{g}_i\}_{i=1}^{K \times R}$=FC($\{\boldsymbol{a}_i\}_{i=1}^{K \times R}$), after which softmax is applied to obtain probability distribution vectors $\{\boldsymbol{s}_i\}_{i=1}^{K \times R}$, i.e., $\{\boldsymbol{s}_i\}_{i=1}^{K \times R}$=Softmax($\{\boldsymbol{g}_i\}_{i=1}^{K \times R}$). All the augmented multi-view features are classified to the corresponding image-level label by minimizing the cross-entropy between $\{\boldsymbol{s}_i\}_{i=1}^{K \times R}$ and the one-hot label vector $\boldsymbol{y}$:

$$L_{local} = -\sum_{i=1}^{K \times R} \sum_{c=1}^{C} y_c log(s_{i,c}), \tag{6}$$

where y_c and $s_{i,c}$ are the c-th element of $\boldsymbol{y}$ and $\boldsymbol{s}_i$ respectively. We call this local loss as the inputs to the main classifier head are multi-view local features. The total loss function is the combination of the global loss and the local loss:

$$L_{total} = L_{global} + L_{local}. \tag{7}$$

We find that the final classification performance is not sensitive to tuning the weights of L_{global} and L_{local} in Eq. 7, and thus assign equal weights for the two loss functions.

At inference time, the final prediction of the image is the ensembled result of the predictions to all the augmented features, i.e., the predicted class label p is:

$$p = \arg\max_c (\sum_{i=1}^{K \times R} g_{i,c}). \tag{8}$$

For computational efficiency, the inputs of the entire model is actually a mini-batch of N images, the complete loss function including the batch dimension is as below:

$$\begin{aligned} L_{total} = &-\frac{1}{N} \sum_{n=1}^{N} \sum_{c=1}^{C} y_c^{(n)} log(f_c^{(n)}) \\ &-\frac{1}{N} \sum_{n=1}^{N} \sum_{i=1}^{K \times R} \sum_{c=1}^{C} y_c^{(n)} log(s_{i,c}^{(n)}), \end{aligned} \tag{9}$$

in which the superscript (n) indicates the n-th image sample in a mini-batch. Last but not least, our MV-FeaAug module has no restriction on the main classifier head. Besides single-layer softmax classifier, the main classifier head could be more complex one with multiple hidden layers, or more robust one like prototype classifier [Yang *et al.*, 2018], we demonstrate this in supplementary materials.

3 Experiments

3.1 Experiment Setup

We evaluate our method on the following object, scene, and action classification datasets: Caltech101, Imagenette, Corel5k, Scene15, MIT Indoor67, Stanford Action40, UIUC Event8, UCMLU, RSSCN7, and AID. The details of these datasets are described in supplementary materials. All the images of these datasets are resized to 224×224. We verify the effectiveness of our MV-FeaAug module by applying it to networks with different CNN backbones. For all CNN backbones, the resolution of the output feature maps is kept to be 14×14, upon which we sample 3×3, 5×5, 7×7, and 9×9 multi-scale local features around each anchor point, i.e., we set $L_R = [3, 5, 7, 9]$. In all experiments, we use the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The initial learning rate is $lr = 0.0001$, the batch size is $N = 64$. We only use very simple data augmentation techniques by firstly flipping images horizontally with 0.5 probability, and then enlarging images to 256×256 followed by randomly cropping back to 224×224. Our model is implemented with TensorFlow and run on a single GeForce GTX 1080 Ti GPU.

3.2 Experiment Results

Firstly, we visually demonstrate that our proposed AdaCAM possesses comparable object localization ability as CAM. To this end, we train a network with a structure of [VGG16 backbone→GAP→single FC layer] to generate CAM; and train another network of [VGG16 backbone→single Conv$_{1\times1}$

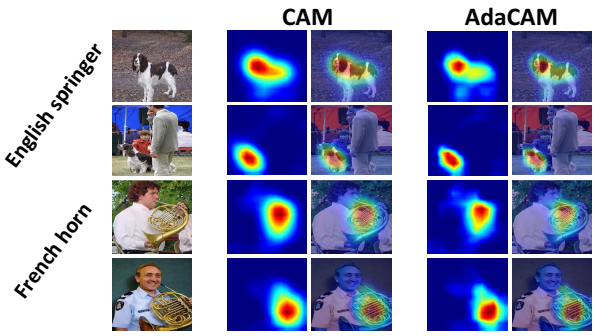

Figure 5: Visual comparison between CAM and our AdaCAM (evaluated on Imagenette validation set) in object localization. Refer to supplementary materials for more results.

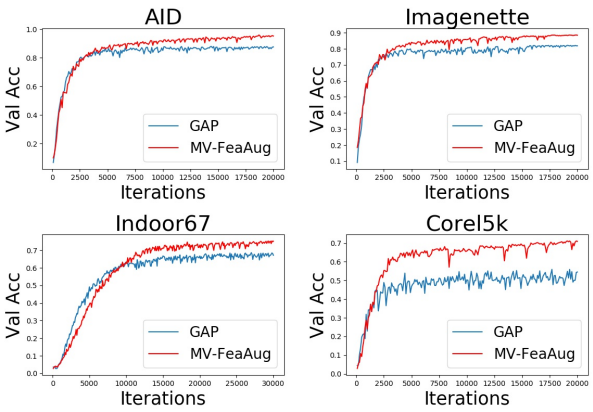

Figure 6: Learning curves comparison between GAP and MV-FeaAug on different datasets. A validation accuracy (Val Acc) is evaluated after every 100 training iterations. The used CNN backbone is VGG16. Refer to supplementary materials for more results.

layer→GAP] to synthesize AdaCAM. As shown in Figure 5, though generated efficiently in a feed-forward manner and independently of class label, the attention maps obtained by our AdaCAM are very close to that of CAM.

We incorporate our MV-FeaAug module to different CNN backbones as a substitute of GAP and achieve consistent and noticeable performance gains. The chosen CNN backbones are VGG16 [Simonyan and Zisserman, 2014], ResNet50 [He *et al.*, 2016], ResNeXt50 [Xie *et al.*, 2017], and MobileNet [Howard *et al.*, 2017]. For 224×224 input images, the output of VGG16 backbone is 14×14 feature maps, upon which we respectively apply GAP and our MV-FeaAug (MFA) module, yielding VGG16_GAP and VGG16_MFA respectively. For ResNet50, ResNeXt50, and MobileNet backbones that contain 5 convolutional stages, the output feature maps are of 7×7 spatial size. We firstly build GAP-based models by applying GAP over them, leading to ResNet50_GAP, ResNeXt50_GAP, and MobileNet_GAP. For evaluation of MV-FeaAug, since 7×7 resolution is too small to sample sufficient multi-view local features, we remove the last convolutional stage of these backbones, yielding the truncated counterparts ResNet41, ResNeXt41, and MobileNet24 that have 14×14 output feature maps, upon which we apply MV-FeaAug module to build ResNet41_MFA, ResNeXt41_MFA, and MobileNet24_MFA respectively. For fair comparison between GAP and MV-FeaAug under the same backbones, we also build ResNet41_GAP, ResNeXt41_GAP, and MobileNet24_GAP by applying GAP over the truncated backbones. We set $K = 50$ and $L_R = [3, 5, 7, 9]$ in our MV-FeaAug module. The used classifier head of all the built models in this section is a simple single-FC-layer softmax classifier, though it is demonstrated in supplementary materials that our module also brings impressive performance gains in the case of more complex classifier head.

We train 20k iterations on Imagenette, Caltech101, AID, and Corel5k, 10k iterations on Scene15, Event8, and RSSCN7, 15k iterations on UCMLU, 30k iterations on Action40, and 40k iterations on Indoor67, making sure that learning converges on all datasets. The validation accuracies achieved by different models on different datasets are reported in Table 1. It is shown that under the same CNN backbone, image classification performance could be improved by a noticeable margin by substituting our MV-FeaAug module for GAP. Furthermore, we demonstrate that GAP based models with a deeper CNN backbone could be surpassed

Model	Datasets									
	Imagenette	Caltech101	Corel5k	Scene15	Indoor67	Action40	Event8	UCMLU	RSSCN7	AID
VGG16_GAP	82.04%	70.42%	54.31%	80.86%	68.25%	52.19%	80.00%	90.95%	91.25%	87.61%
VGG16_MFA	**88.45%**	**76.05%**	**70.74%**	**85.53%**	**75.27%**	**60.36%**	**85.75%**	**96.67%**	**95.29%**	**94.55%**
ResNet50_GAP	83.66%	71.84%	54.55%	81.41%	70.14%	53.43%	81.08%	91.67%	91.43%	89.48%
ResNet41_GAP	82.47%	70.75%	52.72%	80.17%	68.90%	52.56%	79.88%	90.58%	90.54%	88.31%
ResNet41_MFA	**88.60%**	**76.19%**	**69.84%**	**85.00%**	**75.59%**	**60.61%**	**85.25%**	**96.43%**	**94.71%**	**94.89%**
ResNeXt50_GAP	84.10%	72.65%	56.20%	83.18%	71.26%	54.71%	82.95%	92.65%	92.32%	90.24%
ResNeXt41_GAP	82.79%	71.54%	55.66%	81.84%	69.95%	53.50%	81.89%	91.79%	91.18%	88.96%
ResNeXt41_MFA	**88.86%**	**76.78%**	**71.48%**	**86.29%**	**76.38%**	**61.77%**	**87.14%**	**97.31%**	**95.14%**	**95.29%**
MobileNet_GAP	81.26%	70.26%	53.86%	80.24%	67.79%	52.48%	80.50%	90.24%	90.54%	86.87%
MobileNet24_GAP	80.60%	69.78%	51.77%	79.34%	66.96%	51.73%	79.25%	89.37%	89.61%	86.04%
MobileNet24_MFA	**87.55%**	**75.74%**	**69.31%**	**84.97%**	**74.53%**	**59.95%**	**84.75%**	**95.48%**	**93.82%**	**93.52%**

Table 1: Comparison between GAP and our MV-FeaAug (MFA) module with $K = 50$ in validation accuracy based on different CNN backbones. Our MV-FeaAug impressively boosts model performance with simply one more $Conv_{1\times1}$ layer than GAP-based counterpart.

Model	Datasets			Backbone Parameters
	Caltech101	AID	Indoor67	
VGG16_GAP	70.42%	87.61%	68.25%	14.03M
VGG16_SE_GAP	70.96%	88.44%	68.75%	14.18M
VGG16_CBAM_GAP	71.53%	89.37%	69.24%	14.61M
VGG16_SK_GAP	71.25%	89.63%	69.08%	19.55M
VGG16_GC_GAP	71.87%	90.04%	69.51%	14.76M
VGG16_mixup_GAP	71.55%	89.50%	69.67%	14.03M
VGG16_cutout_GAP	70.90%	88.25%	69.14%	14.03M
VGG16_cutmix_GAP	72.10%	90.16%	70.03%	14.03M
VGG16_MFA (ours)	**76.05**%	**94.55**%	**75.27**%	14.03M

Table 2: Comparison of our MV-FeaAug (MFA) with related visual attention modules and data augmentation methods in validation accuracy. We use VGG16 backbone for all the related methods, and set $K = 50$, $L_R = [3, 5, 7, 9]$ in our MV-FeaAug module.

Datasets	GAP	MV-FeaAug				
		K=20	K=30	K=40	K=50	K=60
Imagenette	82.04%	86.27%	87.62%	88.13%	88.45%	88.72%
Indoor67	68.25%	73.51%	74.34%	74.89%	75.27%	75.55%
Action40	53.21%	58.24%	59.36%	59.98%	60.36%	60.68%
UCMLU	90.95%	94.52%	95.33%	96.02%	96.67%	97.05%

Table 3: Study of the impact of K to validation accuracy on different datasets. The backbone network is VGG16.

by our MV-FeaAug based models with a truncated shallower backbone, which shows potential value of our method in model compression. Learning curves of VGG16_GAP and VGG16_MFA evaluated on the validation set of different datasets are shown in Figure 6, our MV-FeaAug module boosts performance consistently on different datasets.

Since our method relates to both attention mechanism and data augmentation, we quantitatively compare our MV-FeaAug with methods of both two fields. Based on VGG16 backbone, we separately attach a SE_block [Hu *et al.*, 2018], a CBAM_block [Woo *et al.*, 2018], a GC_block [Cao *et al.*, 2019] to the end of each convolutional stage of VGG16 backbone, introducing channel-wise attention, channel-wise and spatial attention, global context self-attention to the network respectively, yielding VGG16_SE, VGG16_CBAM, and VGG16_GC backbones. Besides, we replace the last convolutional layer of each convolutional stage of VGG16 backbone with a SK_block [Li *et al.*, 2019] that combines channel-wise attention with dynamic kernel-size selection, yielding VGG16_SK backbone. For SE_block, CBAM_block, and SK_block, we set the reduction ratio $r = 2$. As shown in Table 2, though injecting these attention modules can improve model performance as compared to the baseline model VGG16_GAP, the effects are very limited compared with our MV-FeaAug module (i.e., VGG16_MFA), which could be due to that our method (i) explicitly attends to class-related local features; (ii) augments training samples at feature space by sampling diverse multi-view features, while the compared attention modules do not possess such object localization and data augmentation abilities. Moreover, adding these attention modules increase backbone network parameters, while our method improves performance remarkably without enlarging

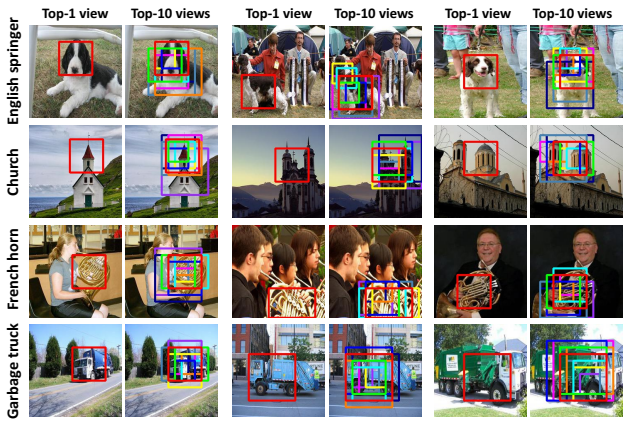

Figure 7: Visualization of the top-1 and top-10 local regions that correspond to local features with the highest prediction confidence of the corresponding image class. Please zoom in for better resolution and refer to supplementary materials for more results.

backbone network. In addition, as Table 2 shows, our method also noticeably surpasses classic image-level data augmentation methods, e.g., Mixup [Zhang *et al.*, 2017], Cutout [DeVries and Taylor, 2017], and CutMix [Yun *et al.*, 2019] (we set α_{mixup}=0.2 and mask size to be 56×56 in these methods), since the augmented multi-view features in our module collaboratively contribute to the final prediction of the image, while these methods do not involve such ensemble learning functionality. Ablation study of the contributing factors of our MV-FeaAug module to classification performance is demonstrated in supplementary materials.

We also explore the influence of K to classification performance based on VGG16_MFA model. Results in Table 3 shows that validation accuracy improves with the increase of K, indicating that sufficient number of the augmented multi-view features is important to model performance.

Based on VGG16_MFA, we select from all the sampled $K \times R$ multi-view local regions top-k ones that correspond to local features with the highest prediction confidence of the corresponding image class in the main classifier head, and map these top-k regions from feature scale to image scale as visualization of our MV-FeaAug. Results of top-1 and top-10 local regions evaluated on images of the Imagenette validation set are shown in Figure 7, we see that the augmented multi-view local features correspond to the selected local regions that basically gather around the image-class-related objects, which demonstrates the effectiveness of our method in extracting rich class-related local features as a kind of attention-based feature-level data augmentation.

4 Conclusion

This paper proposes MV-FeaAug module that performs feature-level data augmentation by sampling multi-view class-related local features with our proposed AdaCAM. By virtue of feature augmentation and multi-view ensemble learning, our module boosts image classification performance noticeably. In future work, we will explore the potential of our method in other learning tasks such as few-shot learning.

References

[Cao *et al.*, 2019] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[DeVries and Taylor, 2017] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[He *et al.*, 2016] 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.

[Howard *et al.*, 2017] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[Hu *et al.*, 2018] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[Huang *et al.*, 2017] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[Iandola *et al.*, 2016] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[Li *et al.*, 2019] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 510–519, 2019.

[Lin *et al.*, 2013] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

[Selvaraju *et al.*, 2017] 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.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[Szegedy *et al.*, 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[Wang *et al.*, 2017] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. Residual attention network for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164, 2017.

[Woo *et al.*, 2018] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[Xie *et al.*, 2017] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[Yang *et al.*, 2018] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Robust classification with convolutional prototype learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3474–3482, 2018.

[Yun *et al.*, 2019] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6023–6032, 2019.

[Zagoruyko and Komodakis, 2016] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[Zhang *et al.*, 2017] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[Zhang *et al.*, 2020] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020.

[Zhou *et al.*, 2016] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.