Multi-Scale Low-Discriminative Feature Reactivation for Weakly Supervised Object Localization

Bo Wang, Chunfeng Yuan, Bing Li, Xinmiao Ding, Zeya Li, Ying Wu, and Weiming Hu

***Abstract*—For weakly supervised object localization (WSOL), how to avoid the network focusing only on some small discriminative parts is a main challenge needed to solve. The widely-used Class Activation Mapping (CAM) based paradigm usually employs Adversarial Learning (AL) strategy to search more object parts by constantly hiding discovered object features, but the adversarial process is difficult to control. In this paper, we propose a novel CAM-based framework with Multi-scale Low-Discriminative Feature Reactivation (mLDFR) for WSOL. The mLDFR framework reactivates the low-discriminative object parts via bottom-up continuous feature maps recalibration and multi-scale object category mapping. Compared with the AL-based methods, our method fully improves the localization power of the network without damaging the classification power and can perform multi-instance localization, which are hard to achieve under the AL-based framework. Moreover, the mLDFR framework is flexible, and can be built on the top of various classical CNN backbones. Experimental results demonstrate the superiority of our method. With VGG16 as backbone, we achieve 46.96% Cls-Loc top1 err and 66.12% CorLoc on ILSVRC2014, 38.07% Cls-Loc top1 err and 75.04% CorLoc on CUB200-2011, surpassing the state-of-the-arts by a large margin.**

***Index Terms*—weakly supervised object localization, feature recalibration, multi-scale class activation mapping**

I. Introduction

THE CNN-based fully supervised frameworks [1], [2], [3], [4], [5], [6], [7] achieve prominent performance in object detection task. However, due to the high cost and ambiguous decision of annotation [8], [9], these technologies have not been well applied in practice. Weakly supervised object localization (WSOL) aims to learn the appearances and locations of objects simultaneously by using only the image-level labels. This technology cannot only significantly reduce the annotation cost, but also helps to learn object characteristics that accord with machine cognition. Therefore, WSOL is attracting more and more attention in recent years. Nevertheless, learning object locations using only image-level labels is a challenging task, since the object instances may appear at different scales and numbers, under different viewpoints, as well as may be occluded and cropped.

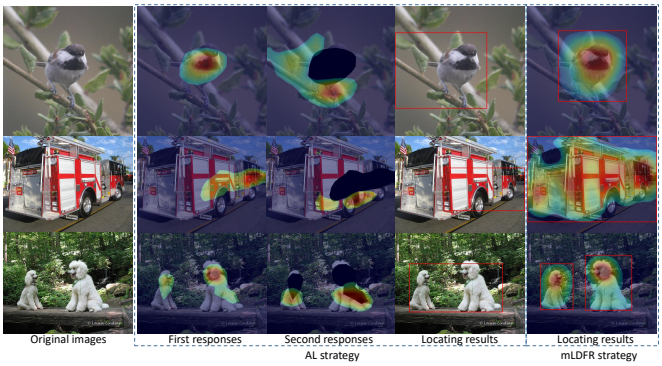


Fig. 1. The examples of AL strategy (middle dotted box) and our mLDFR (right dotted box). Top: For a small instance, hiding operation results in the losing of label-related object instance, causing the network to learn the wrong category features and produce incorrect spatial responses; Middle: For a big instance, a few hiding and discovery operations of the AL strategy cannot capture the full object extent. Bottom: For multi-instances, different response regions are independent of each other, that is, belong to different instances, but the AL strategy combines all responses into only one box.

Zhou et al. [10] have shown that neurons of CNN actually behave as object detectors. Many subsequent studies, e.g. CAM [11], HaS [12], SP [13], ACoL [14], SPG [15], ADL [16], DANet [17], and I2C [18], use neurons to locate objects in the weakly supervised way. They modify the network structure and use the global pooling layer instead of the fully connected layer for feature fusion, e.g. Global Average Pooling (GAP) [11] or Global Max Pooling (GMP) [9]. Then they learn the Class Activation Map (CAM) under this setting to achieve spatial localization of objects. Specifically, there are two kinds of methods to obtain CAM. On one hand, some works (e.g. CAM [11], SP [13]) use a two-step method, which is to project the weights of object categories of the output layer back onto the last convolutional feature maps to generate CAM. On the other hand, in order to improve the integration and computational efficiency, some works (e.g. CAM-GMP [9], ACoL [14]) add a Convolutional Classification Layer (Conv-Cls) on top of the backbone to generate CAM directly, and

Bo Wang is with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, with School of Software and Microelectronics, Peking University, Beijing, China (e-mail: wangbo@ia.ac.cn)

Chunfeng Yuan is with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China (e-mail: cfyuan@nlpr.ia.ac.cn)(Corresponding author)

Bing Li is with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, with PeopleAI, Inc. (e-mail: bli@nlpr.ia.ac.cn)

Xinmiao Ding is with School of Information and Electronic Engineering, Shandong Technology and Business University, Yantai, China (e-mail: dingxinmiao@126.com)

Zeya Li is with Beijing Institude of Tracking and Telecommunications Technology (BITTT), Beijing, China (e-mail: lizeya20082009@163.com)

Ying Wu is with Department of Electrical Engineering & Computer Science, Northwestern University, IL, USA (e-mail: yingwu@eecs.northwestern.edu)

Weiming Hu is with CAS Center for Excellence in Brain Science and Intelligence Technology, with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, and also with School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China (e-mail: wmhu@nlpr.ia.ac.cn)

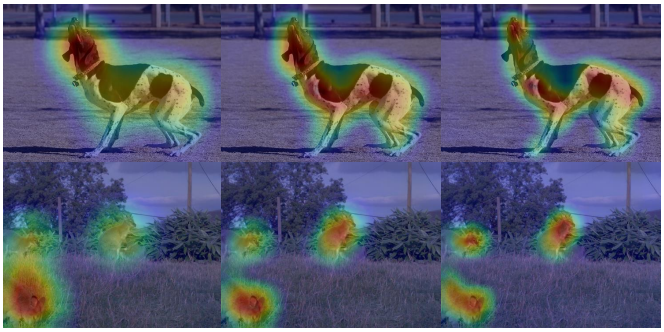


Fig. 2. The activation maps at multiple image scales (224,336,448). For the same model and same image, if the input image scales are different, the activation regions are also different.

perform global pooling on CAM to output image prediction.

The biggest challenge of CAM-based paradigm for localization is: CNN tends to capture the most discriminative features of object to reduce classification ambiguity, resulting in the inability to activate the complete spatial responses of object on CAM. Many existing approaches (e.g. HaS [12], TPL [19], ACoL [14]) employ Adversarial Learning (AL) strategy to activate low-discriminative object features discarded by the network. They work in an iterative manner under the constraints of classification performance, that is, continuously hiding the current object response to stimulate network to discover more object regions, and finally fusing all the responses for localization. Although they have the ability to find more object parts, the search and fusion process of this strategy is difficult to control, and it is likely to deteriorate the network's classification performance [14], [16]. We find that AL-based framework mainly has three obvious bottlenecks for weakly supervised localization task, as shown in Figure 1. First, hiding operation may result in the losing of the label-related object instance in the image, causing the network to learn the wrong category features and produce incorrect spatial responses. Second, it is difficult to set the repeat times of searching. Maybe only one searching can find all the object parts, or maybe multiple hiding and discovery operations are needed to capture the full object extent. Third, in a multi-instance scenario, the network can find multiple object instances in different positions, but the fusion operation of AL-based method will merge all instances into only one box, resulting in serious localization errors.

In this paper, we re-analyze the problem of CNN falling into a local discriminative part of the object, and find that it is caused by three reasons. First, due to the insufficient data and poor diversity in the dataset, the appearance features of some categories are very similar, such as the fur of animals, the shells of vehicles. In order to ensure the classification performance, CNN has to discard those features which are low discriminative and easy to be confused. Second, the perception range of neurons is limited, even the top neurons can only perceive a part of the image [10]. As a result, the network may not be able to learn the complete object features. Third, the discriminative features of an object are different at different image scales, so just taking the network default scale as input can only activate a limited number of object features.

Inspired by these reasons, we propose a novel CAM-based framework with Multi-scale Low-Discriminative Feature Reactivation instead of AL strategy for better WSOL. We call it mLDFR framework, which reactivates the low-discriminative object features through bottom-up continuous feature maps recalibration and multi-scale object category mapping. The overall architecture of mLDFR framework is illustrated in Figure 3. Its novelties are mainly two-fold.

On the one hand, we propose a new feature recalibration strategy to solve the problem of incomplete object response. Specifically, we insert feature recalibration modules after the convolution layer and the pooling layer in the backbone. The recalibration modules use global context information to activate low-discriminative features related to objects at each layer, so that the network can maintain complete and strong spatial responses of objects during the forward propagation. Theoretically, any technique that uses global information to transform feature maps can be used as a recalibration module. In our specific implementation, we design a new self-attention (SA) module to recalibrate the pooling feature maps. The traditional self-attention modules (e.g. SE [20], GE [21], BAM [22], CBAM [23]) are mainly used to recalibrate convolutional features, with the purpose of improving classification performance. Our self-attention module is used to recalibrate pooling features, and the purpose is to improve the localization performance. For convolutional feature maps, we employ the Batch Normalization (BN) module [24] to recalibrate them, which has been widely used in network architecture. Although BN [24] was previously proposed as a regularization technique for accelerating and stabilizing training, we revisit it from the perspective of feature recalibration and find that it recalibrates the features of a certain visual pattern in the spatial domain by using cross-instance global spatial context information.

On the other hand, we propose a Multi-scale Class Activation Mapping (MsCAM) technique to capture different and complementary features of object by fusing CAMs of multi-scale images. This technique is inspired by the question: whether the discriminative parts that the network relies on for recognition are consistent for objects of different scales of the same category in the dataset. Since the receptive field of the network is theoretically fixed, that is, for objects of different scales, the parts that the network can perceive are different. For a small-scale object, the model can capture complete object features, but for a large-scale object, the model can only perceive a local part of the object. So it can be inferred that for objects of different scales, the discriminative features captured by the model should be different. The discriminative part of one scale may lose the discriminability at other scales. We visualize the CAM of the same image at different scales, as shown in Figure 2. For the same model and same image, if the input image scales are different, the activation regions are also different. For example, we firstly observe dog images in the first row of Figure 2. At the 224 scale, only the head gets strong responses. At the 336 scale, the abdomen also gets strong responses. At the 448 scale, the responses of the head and abdomen are reduced, while the tail region gets attention. Fusing the CAMs of these three scales will surely lead to better localization performance. Combining activation maps

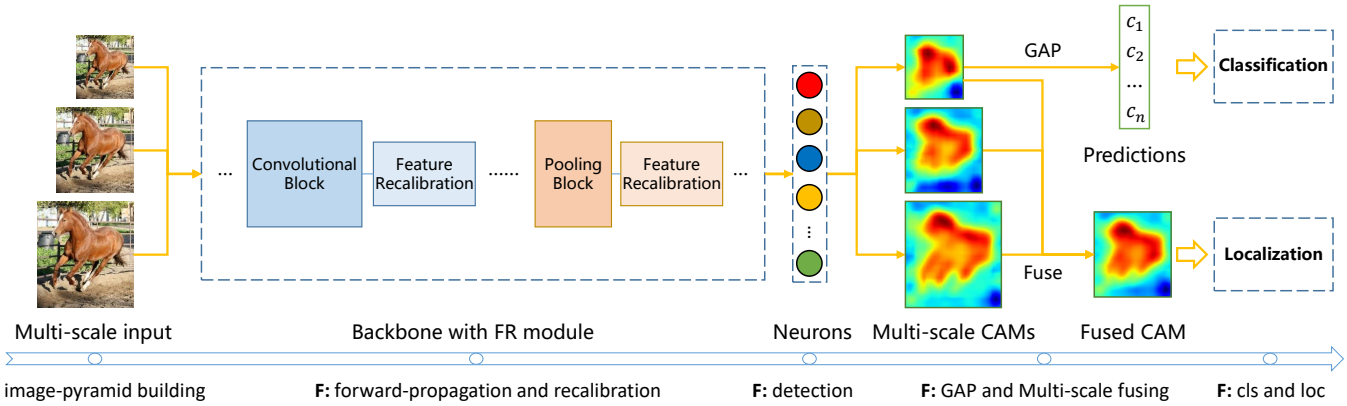


Fig. 3. The Architecture of Multi-scale Low-Discriminative Feature Reactivation (mLDFR) for weakly supervised object localization.

of multiple image scales also helps multi-instance localization. For example, in the second row of Figure 2, only the rabbit in the left bottom at the 224 scale has a high response, while at the 336 and 448 scales, the other two rabbits in the image also get enough attention from the network. In addition, it can be seen from the two examples that the network responds more finely to objects on a large scale, which helps to achieve more precise localization. Zhang et al. [25] used a multi-scale fusion similar to ours in test phase of weakly supervised semantic segmentation task. But for the first time, we analyze the effect of multi-scale technique theoretically and experimentally on improving the object responses. The MsCAM can be used as a standard post-processing step for subsequent WSOL work.

Our mLDFR framework has two advantages. First, it has strong extensibility, and it is easy to build an mLDFR network based on the standard CNNs. Second, it avoids fusion operations and can perform multi-instance localization in complex scenes under the CAM-based paradigm.

In summary, our main contributions are as follows.

- We propose a novel Multi-scale Low-Discriminative Feature Reactivation (mLDFR) framework to tackle WSOL problem. Instead of adopting adversarial learning strategy, the proposed method captures more object regions through feature recalibration strategy and multi-scale activation, avoiding the contradiction between the classification and localization power of the network. In addition, our method can achieve multi-instance localization.
- We design a new self-attention (SA) module to recalibrate pooling features and employ the Batch-Normalization (BN) module to recalibrate convolutional features. In the test stage, we propose a Multi-scale Class Activation Mapping (MsCAM) technique for localization task to capture more complementary object features and merge them to locate the complete object extent.
- Our mLDFR is flexible and we instantiate it with two backbones VGG16 and GoogLeNet. With VGG16 as backbone, our method achieve 46.96% Cls-Loc top1 err and 66.12% CorLoc on ILSVRC2014, 38.07% Cls-Loc top1 err and 75.04% CorLoc on CUB200-2011, surpassing the state-of-the-arts by a large margin.

The code and models of our paper are released at: https://github.com/wangbo2016/wsol-mLDFR-2020.

II. Related Work

Object detection and segmentation have always been hot topics in the field of machine learning. During the development process, a variety of excellent methods have emerged, such as *Local feature* model [26], [27], [28], [29], *Active contours* model [30], [31], and *Deep learning* model [3], [4], [5], [32], [33]. Although the object detection and segmentation technologies have made great progress, the cost of building the corresponding system in practical is very high. Such intelligent systems usually require large-scale labeled data and a lot of time to debug parameters. Therefore, many works study how to reduce the cost of building intelligent systems, such as *Weakly supervised learning* [34], [35], [36], *Instance selection* [37], [38], etc. The Weakly Supervised Object Localization (WSOL) studied in this work is an important research field for reducing the cost of intelligent systems. Next, the technologies related to this paper is briefly reviewed.

A. Weakly Supervised Object Localization

MIL-based Paradigm. MIL-based paradigm includes two stages [39], [40], [41], [42], [43], [44], [45], [46], [47], [48], [49]. It first uses the Region Proposals methods such as SelectiveSearch (SS) [50] or EdgeBoxes (EB) [51] to generate lots of object proposals, and then selects the one that contributes the most to image-level classification scores as object bounding box. Because of its heavy reliance on the quality of Region Proposal methods and high computational costs, the MIL-based methods are difficult to perform object detection directly in a weakly supervision way. Therefore, MIL-based paradigm is often used for mining pseudo object-level labels, and then the generated pseudo object annotations are used to train the fully supervised detector.

CAM-based Paradigm. CAM-based paradigm is an end-to-end pipeline [9], [11], [13], [19], [12], [52], [14], [15], [16], [17], formally proposed by [11] for the first time. This kind of methods use the global pooling layer instead of the fully connected layer for feature fusion, thereby liberating the localization ability of neurons. In essence, CAM-based paradigm treats neurons as detectors that predict the locations of objects directly during the forward propagation of the CNN. Under the CAM-based paradigm, most works adopt an adversarial learning strategy to solve the problem of insufficient

object activation. Singh and Lee [12] propose a Hide-and-Seek technique that forces the network to seek more object-related parts by randomly hiding grid regions in the training image. Kim et al.[19] and Zhang et al. [14] set multiple adversarial classifiers and use semantic information to guide the erasing process, which result in discovering new and complementary object parts more effectively. Choe and Shim [16] improves dropout layer by introducing a attention mechanism that integrates erasing and discovering operations into a single classifier, thereby improving processing efficiency. In addition, Zhang et al. [18] propose an I2C method that significantly improves the quality of object localization maps by forcing the object features of the same categories to be consistent. Zhang et al. [53] split the WSOL task into two parts: the class-agnostic object localization and the object classification. This method solves the problem that classification and localization are difficult to balance under the traditional CAM framework. Choe et al. [54] conduct an in-depth analysis of the definition and challenges of the WSOL task, and propose some new evaluation protocols and a new benchmark dataset.

CAM-based VS MIL-based. In the localization process, CAM-based paradigm faces the complete and huge image space without any prior knowledge. In contrast, since the SS or EB method can recall most bounding boxes of groundtruth in a small set of region proposals, the MIL-based paradigm is actually searching in a smaller and higher-quality search space. However, the CAM-based methods can achieve classification and location in a forward propagation simultaneously, so it is faster and computationally cheaper.

B. Feature Recalibration

The recent works have focused on improving the performance of CNN by optimizing the internal features of network, especially the self-attention mechanism [55], [56], [57], [20], [21], [22], [58] and normalization technology [24], [59], [60], [61], [62]. Although the research motivation of these technologies are different, their calculation process can be seen as generating the recalibration weights to transform the original feature maps, thereby enhancing representation ability of the network, so in this work we regard them all as a kind of feature recalibration technology.

Self-Attention mechanism. The self-attention mechanism helps to capture the internal correlation of features from a global perspective and generate a set of importance recalibration weights, thereby strengthening the category-related features. The self-attention module mainly recalibrates the importance of features from three domains, including spatial domain, channel domain, and mixed domain (space + channel). Jaderberg et al. [55] propose a Spatial Transformer module, which learns spatial transformation weights to warp object features. Wang et al. [56] build a Residual Attention Network by using a self-attention module with an encoder-decoder structure, which improves CNN's classification performance and saves parameter overhead. Hu et al. [20] propose a SE block, which recalibrates channel-wise features by modelling interdependencies between channels. Park et al. [22] propose a *Bottleneck Attention Module* (BAM) that generates a 3D attention map using the information of channel and spatial domains simultaneously. Woo et al. [23] propose a *Convolutional Block Attention Module* (CBAM) similar to BAM [22], in which the bottleneck structure is only used in the channel attention sub-module. In this paper, we also design a self-attention module to generate recalibration weights from the channel and the spatial domains. Unlike BAM[22] and CBAM[23] which are used to recalibrate convolutional features, our self-attention module is used to recalibrate pooling features.

Normalization. Normalization technologies are usually used to regularize the model parameters, thereby improving the efficiency and stability of the training, and the generalization ability of the model. The essence of these technologies are to use the global context information to recalibrate the activation distribution of the feature maps. The main difference between different normalization technologies lies in the definition of the global scope. Ioffe and Szegedy [24] propose BatchNorm for large-scale image classification, which uses specific channel information of all instances in a mini-batch to learn recalibration weights. Ba et al. [59] propose LayerNorm for single-instance training scenarios, and use all channel information of a single hidden layer to perform recalibration. Ulyanov et al. [60] propose InstanceNorm, which uses only one information channel, further narrowing the scope of information statistics for normalization operations and has better performance in image generation tasks [63], [64], [65]. Wu and He [61] group the channels of a convolutional layer in CNN and make statistics within the grouping range, which performs better in scenarios that require a smaller batch-size. Since WSOL is based on large-scale image classification, we employ BatchNorm [24] as the recalibration module in the framework.

III. METHOD

We first revisit the CAM technology and show that the CAM-based paradigm is essentially to train the class-specific neuron detectors. Then, we elaborate the details of the proposed Multi-scale Low-Discriminative Feature Reactivation (mLDFR) framework, including the design of the recalibration module and the multi-scale object category mapping.

A. Revisiting Class Activation Mapping

In this section, we briefly revisit the CAM technology proposed by Zhou et al. [10] and two methods for generating CAM. Then, we analyze mathematically why the essence of CAM is to train a group of neurons as object detectors. Zhou et al. [10] proved experimentally that object detection is an important part in the process of building representation by the network. This means that neurons trained by classification task have their own semantic preferences, which can be used as object detectors. This work laid the foundation for CAM-based object localization. In the development of CAM technology, there are mainly two methods to generate CAM, one-step and two-step, corresponding to two network architectures. Zhang et al. [14] made an analysis of this, here we briefly review it.

Specifically, for a backbone and a given image, the feature maps of last convolutional layer are denoted as $f \in \mathbb{R}^{K\times H\times W}$, where $H \times W$ represents the spatial resolution and K is the number of channels. In [11], [13], [52], a Global

Average Pooling (GAP) layer is added before a FC layer on top of the backbone, in which the FC layer has C neurons (corresponding to the number of categories). Firstly, the feature maps f are fed into the GAP layer to pool as image representation, denoted as $F_k = \frac{\sum_{x,y} f_k(x,y)}{H \times W}$, where $f_k(x,y) \in \mathbb{R}$ is the activation of kernel k of the last convolutional layer at spatial location (x,y). Then, the pooled features F are fed into the FC layer to output the category scores, which can be defined as:

$$\begin{aligned} y_c^{fc} &= \sum_k w_{c,k}^{fc} F_k \\ &= \sum_k w_{c,k}^{fc} \frac{\sum_{x,y} f_k(x,y)}{H \times W} \\ &= \frac{1}{H \times W} \sum_{x,y} \sum_k w_{c,k}^{fc} f_k(x,y), \end{aligned} \tag{1}$$

where $\boldsymbol{w_c^{fc}} \in \mathbb{R}^K$ is the weight vector of the FC layer corresponding to category c. Finally, by projecting the weights of FC layer back to the feature maps f, the class activation map A_c^{fc} for category c can be obtain:

$$A_c^{fc}(x,y) = \sum_k w_{c,k}^{fc} f_k(x,y). \tag{2}$$

For better integration and efficiency, [14], [17] add a Classification Convolutional layer (called Conv-Cls layer) followed by a GAP layer on top of the backbone, in which the Conv-Cls layer is set as: output_channels=C, kernel_size=1×1, stride=1, bias=False. In this architecture, the class activation map A_c^{conv} can be generated directly by the Conv-Cls layer as follows:

$$A_c^{conv}(x,y) = \sum_k w_{c,k}^{conv} f_k(x,y), \tag{3}$$

where $\boldsymbol{w_c^{conv}} \in \mathbb{R}^{K\times1\times1}$ is the weight matrix of Conv-Cls layer corresponding to kernel c, which can also be seen as the weights of category c. Then, after the calculation of GAP layer, the prediction is:

$$\begin{aligned} y_c^{conv} &= \frac{1}{H \times W} \sum_{x,y} A_c^{conv}(x,y) \\ &= \frac{1}{H \times W} \sum_{x,y} \sum_k w_{c,k}^{conv} f_k(x,y). \end{aligned} \tag{4}$$

By comparing Equation 1 and Equation 4, we draw two conclusions: 1) If the network uses GAP to pool the feature maps, the two CAM generating methods are mathematically equivalent, as proved by Zhang et al. [14]. 2) As we can see, $\boldsymbol{w_c}$ is the weights of a neuron used to recognize category c, and $f(x,y)$ is the features of spatial location (x,y) in the last convolutional feature maps f. Considering that the convolution calculation is similar to the sliding window technique, the generation of CAM can be seen as a procedure in which neurons identify the category of each location in the feature maps. Based on works [66], [10], the receptive field of a neuron corresponds to a region in the image. Therefore, the CAM-based WSOL paradigm actually trains C neurons (corresponding to C object categories) to detect object instances in the image. We call them Neuron Detectors. In this paper, we follow the second CAM generation method.

B. The proposed mLDFR framework

To reactivate the low-discriminative features discarded during the forward propagation, we propose to add feature recalibration modules to the CAM-based pipeline. The recalibration modules use the global context information to model the interdependencies between the features, thereby using high-discriminative features to strengthen the low-discriminative features related to the object. Specifically, we design a self-attention (SA) module to recalibrate the pooling feature maps. At the same time, we revisit the Batch-Normalization (BN) [24] from the perspective of spatial recalibration and insert it into the network to recalibrate the convolutional feature maps.

Pooling features recalibration. The self-attention mechanism uses the global context information of the channel and spatial domains to enhance the recognition capability of the network. Channel attention reinforces specific visual patterns related to the object category by explicitly modelling the semantic relationship between channels [20]. Spatial attention captures the dependencies between local features in the spatial domain to strengthen the importance of low discriminative features related to the object [67]. In this work, we design a self-attention (SA) module that sequentially learns the channel attention and spatial attention, as shown in Figure 4.

To generate the channel attention, we use Global Average Pooling (GAP) to aggregate spatial information because GAP encourages the network to focus on the full extent of object, which is more conducive to box localization [11]. First, we use GAP to aggregate the input feature maps, generating a channel descriptors $F_{avg}^c \in \mathbb{R}^K$. Second, we feed the descriptors into a fully connected (FC) layer to model the dependencies between channel-wise features, generating the attention scores. Then, we use the Sigmoid function to generate the normalized recalibration weights of the channel.

$$Att_{\text{channel}} = \sigma\left(W^{fc}\left(F_{\text{avg}}^c\right)\right), \tag{5}$$

where σ is the Sigmoid function, $W^{fc} \in \mathbb{R}^{K\times K}$ represents the weights of the FC layer. To generate the spatial attention, first, we calculate the mean and maximum value of the feature map along the channel dimension to obtain two spatial descriptors $F_{avg}^s \in \mathbb{R}^{H\times W}$ and $F_{max}^s \in \mathbb{R}^{H\times W}$, which aggregate the channel information from different perspectives. Second, we concatenate them and feed the result $[F_{\text{avg}}^s, F_{\text{max}}^s] \in \mathbb{R}^{2\times H\times W}$ into a convolutional layer to highlight the spatial response of the object. Then, we use the Sigmoid function to generate the normalized spatial recalibration weights,

$$Att_{\text{spatial}} = \sigma\left(W^{conv}\left(\left[F_{\text{avg}}^s, F_{\text{max}}^s\right]\right)\right), \tag{6}$$

where σ is the Sigmoid function, $[\cdot,\cdot]$ represents the concatenation operation, and W^{conv} is the weights of the convolution operation. When inserted into the backbone, the convolutional layer of the spatial attention sub-module needs to be adjusted according to the insertion position. At the bottom of the network, a large-size convolution kernel is used, and at the top of the network, a small-size convolution kernel is used. This setting is more conducive to fine localization of object features (the activation areas match the object parts more). Specifically, we add the proposed SA modules after the 2nd,

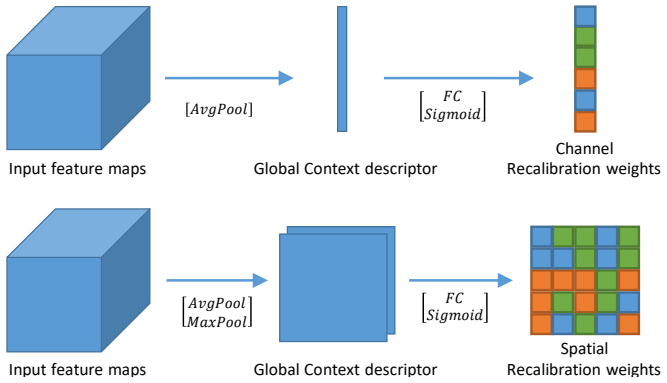


Fig. 4. Illustration of our self-attention module. In channel dimension, we use GAP to aggregate spatial information. In spatial dimension, we use max-pooling and avg-pooling to aggregate channel information.

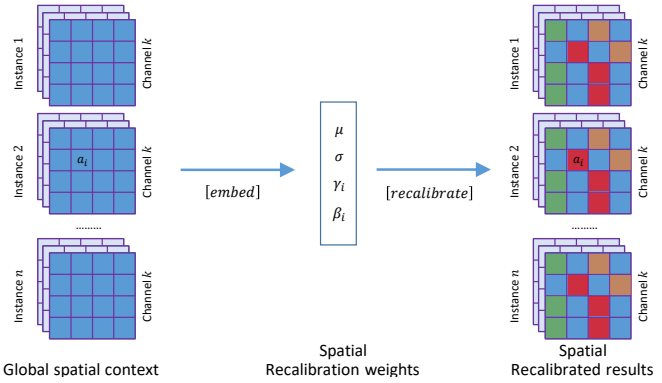


Fig. 5. BatchNorm uses cross-instance global spatial context information to learn the recalibration weights, and uses the weights to strengthen or weaken feature responses.

3rd, and 4th pooling layers of the backbone, corresponding to the convolution kernel sizes of 7x7, 5x5, and 3x3, respectively.

Finally, we use two dimensions of attention weights to recalibrate the feature maps. For an input feature map $X \in \mathbb{R}^{K\times H\times W}$, we sequentially perform features recalibration in the spatial domain and the channel domain.

$$\tilde{X} = FR_s\left(Att_{\text{spatial}}, FR_c\left(Att_{\text{channel}}, X\right)\right), \tag{7}$$

where FR_c refers to the channel-wise feature recalibration and FR_s refers to the spatial-wise feature recalibration. The self-attention module models the correlation between features in the global scope, helping the network use high discriminative features to strengthen low discriminative features of the object. especially at the bottom layer of the network.

Convolutional features recalibration. BN was previously proposed as a regularization method to alleviate the problem of gradient disappearance and improving the training speed by adjusting the input distribution of each layer to a standard normal distribution with the mean of 0 and variance of 1. We revisit BN [24] from the perspective of feature recalibration and find that it actually recalibrates the spatial importance distribution of object features very well. Specifically, for an activation a_i, BN module performs: 1) calculate the mean μ and variance σ to normalize the feature maps, and 2) learn a pair of factors γ_i and β_i to restore the representation power of the network. It can be expressed as follows:

$$\begin{aligned} a_i^{norm} &= \gamma_i \cdot \frac{a_i - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta_i \quad i \in M, \\ \mu &= \frac{1}{M}\sum_{i=0}^{M} a_i, \\ \sigma^2 &= \frac{1}{M}\sum_{i=0}^{M}\left(a_i - \mu\right)^2, \end{aligned} \tag{8}$$

where ϵ is a constant, M is the number of neurons in the k-th channel for all instances in a batch. The calculation process of BN can be regarded as the use of cross-instance global spatial context information to generate the recalibration weights of neurons and activate the low-discriminative neurons related to object, as shown in Figure 5. Specifically, in the training process for image classification task, the convolution kernel gradually forms its own visual pattern preference [10], that is, the feature map of a channel usually corresponds to a specific visual pattern. From this perspective, the BN module essentially recalibrates the features of a certain visual pattern in the spatial domain. Despite the apparent simplicity of BN, the network-BN can achieve amazing improvements in localization performance with the help of MsCAM technique.

C. Multi-scale Class Activation Mapping

Since CNN's response to objects on a single scale is insufficient, we propose a Multi-scale Class Activation Mapping (MsCAM) based on image pyramid to solve this issue. Figure 6 shows the localization process based on MsCAM.

Specifically, we use a set of scaling factors to build the image pyramid and feed the scaled images into the trained locating model. Our network is a fully convolutional structure, and it can process images of any resolution. Then, CAMs of multi-scale are resized to the original image size and summed linearly to generate final CAMs, denoted as $A_c^{final} \in \mathbb{R}^{C\times H\times W}$. Compared to AL-based methods, our method only needs to train one classifier and does not require fine training control. In order to obtain the segmentation map $S_c^{final} \in \mathbb{R}^{C\times H\times W}$, we first normalize A_c^{final} to $0-1$, and then remove the regions where the activation value is less than the threshold. The process can be defined as follows,

$$S_c^{final} = \left(\frac{A_c^{final} - \min\left(A_c^{final}\right)}{\max\left(A_c^{final}\right) - \min\left(A_c^{final}\right)}\right) - t \tag{9}$$

where t is the segmentation threshold. Finally, we generate bounding box on each connected region, and each bounding box corresponds to an object instance. The scaling factors of the pyramid are not fixed, but depend on the characteristics of the dataset. MsCAM make good use of cross-scale recognition capability of neurons, which can be used as a standard post-processing step for subsequent WSOL work.

D. Classification and Localization based on mLDFR

The proposed mLDFR framework is easy to implement based on the existing network architectures. Given a classification backbone, we first modify it to be a localization network,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2021.3091833, IEEE Transactions on Image Processing

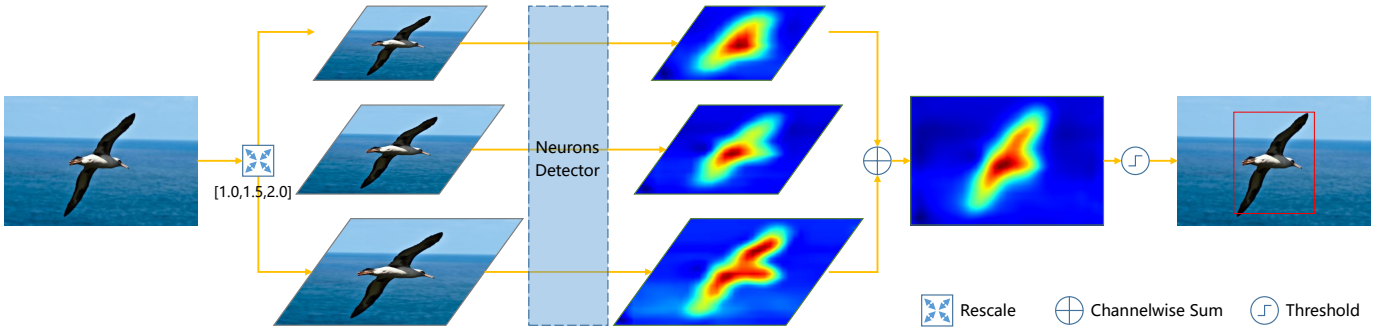


Fig. 6. Multi-scale Class Activation Mapping (MsCAM) for weakly supervised object localization.

and then insert the recalibration modules after the convolution module and the downsampling module. Specifically, we remove the last downsampling module of backbone to obtain the feature maps with higher resolution. This modification is intuitive because the higher the feature maps resolution, the better the precise of object localization. Note that except for VGG16 backbone [68], the way we modify the backbone is different from the traditional CAM-based method [11], refer to Section IV-B for details. Next, we insert a BN module after each convolutional module to recalibrate the convolutional feature maps and insert a SA module after each downsampling module to recalibrate the compressed feature maps. Finally, we add a Conv-Cls layer to generate CAM, followed by a GAP layer to output image prediction.

A specific mLDFR-based localization network instantiated with VGG16 as the backbone is illustrated in Figure 7. In the training stage, we use the softmax cross entropy loss to guide the learning of the network, as follows.

$$\text{loss}\left(y^{pre}, y^{gt}\right) = -\sum_{c}^{C}\left(y_c^{gt} \cdot \log y_c^{pre}\right) \tag{10}$$

where $y^{pre} \in \mathbb{R}^C$ is the network outputs and $y^{gt} \in \mathbb{R}^C$ is the groundtruth label. The mLDFR framework does not forcibly interfere with the learning process of the network. Therefore, through continuous optimization, the classification and localization capabilities of the network can be simultaneously improved. In the testing stage, we use the default scale of the network as input to perform classification task, and use MsCAM technique to perform localization task.

IV. Experiment

We evaluate the weakly supervised object classification and localization performance of the proposed method on popular ILSVRC2014 [69] and CUB200-2011 [70] datasets.

A. Datasets and evaluation metrics

Datasets. ILSVRC2014 [69] is a large-scale dataset of 1000 object categories, consisting of 1.3 million training images, 50k validation images, and 100k test images, with an average of 1 category and 1.2 instances per image, and 89.6% of the images contain 1 instance. CUB200-2011 [70] is a fine-grained dataset of 200 bird categories, consisting of 5994 training images, 5794 test images, with an average of 1 category and 1 instance per image.

Metrics. For ILSVRC2014 and CUB200-2011, we use the Error Metric (top1 error, top5 error) for Classification task (Cls) and Classification-Localization task (Cls-Loc) suggested by [69]. We use CorLoc [71] to measure the pure bounding box localization performance of the model.

B. Implementation details

Network Modification. We validated the proposed mLDFR framework on two popular backbones, including VGG16[68] and GoogLeNet [72]. Specifically, for VGG16 [68], we first remove the last pooling layer and subsequent fully connected layers. Second, we add one convolution layers (output_channels=1024, kernel_size=3×3, stride=1, padding=1, bias=True) to compensate for the loss of learning capacity caused by the removal of the fully connected layers. Then, we insert BN module after each convolutional layer and SA module after the 2nd, 3rd and 4th pooling layer. Finally, we add a Conv-Cls layer (output_channels=C, kernel_size=1×1, stride=1, padding=0, bias=False) followed by a GAP layer. For GoogLeNet [72], instead of removing the last pooling layer and subsequent inception blocks [14], [17], which would reduce the learning capacity of the network, we replace the last pooling layer with a convolution layer (defined as transition layer) (output_channels=832, kernel_size=1×1, stride=1, padding=0, bias=True) to align the channel number between the previous and subsequent convolutional layers. Then, we insert BN module after each convolutional layer, and insert SA module after the 2nd, 3rd and 4th downsampling layer.

To verify the compatibility of the mLDFR framework to various types of backbones, we also conduct experiments on InceptionV3 [73], ResNet50 [74], DenseNet161 [75], and MobileNetV2 [76] backbones. Since the above backbones already contain BN modules, we only add the proposed self-attention module to these backbones. Specifically, we first replace the last downsampling block of the backbone with a transition layer (as before). Take InceptionV3 backbone as an example, the transition layer is set to: output_channels=1280, kernel_size=1×1, stride=1, padding=0, bias=True. This layer is used to to transform the number of channels from 768 to 1280. Then, we insert the proposed SA modules after the 2nd, 3rd and 4th downsampling blocks of the backbone. Finally, we add a Conv-Cls layer followed by a GAP, as before.

For other backbones except VGG16, our modification is different from the traditional approach, which usually removes the last downsampling block and all subsequent layers (such as

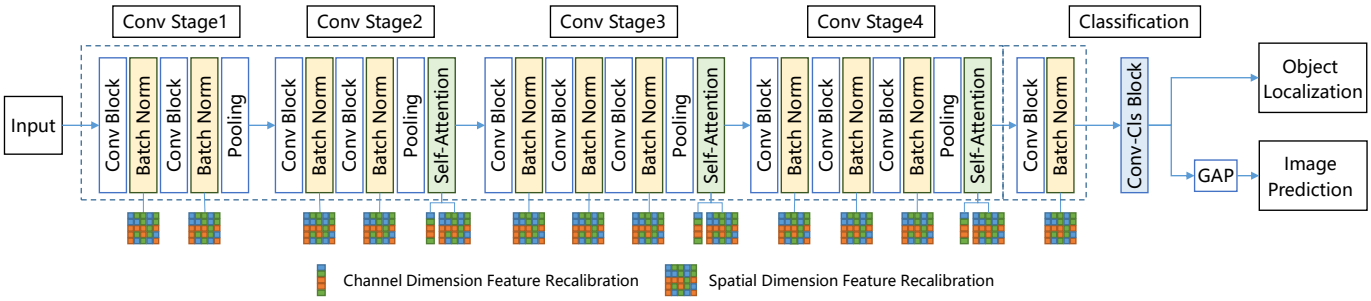


Fig. 7. The mLDFR network based on VGG16 backbone, which uses our Self-Attention module and Batch-normalization [24] module for feature recalibration.

Mixed_7a-7c of InceptionV3, layer4 of ResNet50, feature14-18 of MobileNetV2), and then adds some convolution layers. Because for these network backbones, there are many convolutional layers after the last downsampling block. If these convolutional layers are directly deleted, the learning ability of network will be greatly damaged. Our modification results in the feature maps with higher resolution. Meanwhile, the learning capacity of the network will not be seriously affected. All upgraded networks are pre-trained on ILSVRC.

Training. During learning, we randomly crop the image to different sizes (0.8-1.0) and aspect ratios (0.75-1.33), and then resize the cropped patch to the default input size of the backbone. we use SGD optimizer with an initial learning rate 0.03, momentum 0.9 and weight decay 0.0001. The learning rate is decreased by a factor of 10 if the loss has stopped improving in 3 epochs. The batch-size is set to 128 for DenseNet161, and 256 for other backbones. The number of training epochs is 50 for all datasets, and the model of the last iteration is selected for testing. We train the networks on NVIDIA GeForce RTX 2080 Ti GPU with 11GB memory. Note that the platform we used in all experiments is PyTorch.

Testing. During testing, in the preprocessing stage, we resize the short side of the image to the default input size of the network, and resize the long side in proportion. Then, for the classification task, the image scaled to the default size is fed into the trained network to calculate the classification scores. For the localization task, we use the proposed MsCAM technique to locate objects.

Threshold. In our experiment, we randomly select 10% of the data in the training set for the threshold selection, including the number of pyramid layers and the segmentation threshold. For our experimental benchmarks, we unify the threshold setting, that is, combine the object responses of three continuous scales, i.e. 1.0, 1.5, 2.0 (e.g. 224, 336, 448 for VGG16) and set segmentation threshold to 0.2.

C. Comparisons with the state-of-the-arts

We compare our method with several classical WSOL methods under CAM-based paradigm, which are named as their backbone networks adding their proposed methods.

Classification. To fairly evaluate the classification performance of the mLDFR framework for WSOL task, we compare it with other works under the same experimental setting, including backbones, datasets and evaluation protocols. Table I and Table II show the Top-1/Top-5 classification error on ILSVRC2014 and CUB200-2011, respectively. On

TABLE I
CLASSIFICATION (CLS) ERRORS ON ILSVRC2014.

Method	top-1 err	top-5 err
GoogLeNet [72]	31.90	11.30
GoogLeNet-GAP [11]	35.00	13.20
GoogLeNet-ACoL [14]	29.00	11.80
GoogLeNet-DANet [17]	27.50	8.60
GoogLeNet-mLDFR(Ours)	**26.66**	**8.29**
VGG16 [68]	31.20	11.40
VGG16-GAP [11]	33.40	12.20
VGG16-ACoL [14]	32.50	12.00
VGG16-ADL [16]	30.52	–
VGG16-I2C [18]	30.60	10.70
VGG16-NLCCAM [77]	27.70	–
VGG16-mLDFR(Ours)	**24.79**	**7.32**

TABLE II
CLASSIFICATION (CLS) ERRORS ON CUB200-2011. THE UNDERLINED ACCURACIES INDICATE THAT THE SCORES ARE RETESTED BY US.

Method	top-1 err	top-5 err
GoogLeNet [72]	30.15	10.03
GoogLeNet-GAP [11]	35.40	13.76
GoogLeNet-DANet [17]	28.80	9.40
GoogLeNet-mLDFR(Ours)	**23.68**	**5.22**
VGG16 [68]	23.13	6.20
VGG16-GAP [11]	23.40	7.50
VGG16-ADL [16]	34.73	–
VGG16-ACoL [14]	28.10	–
VGG16-SPG [15]	24.50	7.90
VGG16-DANet [17]	24.60	7.70
VGG16-NLCCAM [77]	26.60	–
VGG16-mLDFR(Ours)	**19.30**	**4.19**

the ILSVRC2014, VGG16-mLDFR (VGG16 as backbone) reports 24.79%/7.32 top-1/top-5 Cls err and GoogLeNet-mLDFR (GoogLeNet as backbone) reports 26.66%/8.29% top-1/top-5 Cls err, which exceed all other WSOL methods and even original networks. On the fine-grained CUB200-2011, our method significantly improves the weakly supervised classification performance. It can be seen that the mLDFR framework can maintain or even improve the classification performance of the backbone while realizing the weakly supervised object localization function. It demonstrates that the mLDFR framework did not destroy the high-discriminative parts of the feature maps during the recalibration process. At the same time, the activated low-discriminative features further improve the network's representation ability.

Classification with Localization. The Classification with Localization task (Cls-Loc) [69] requires that the model can not only correctly predict the image category, but also locate an object region that overlaps over 50% with any of the ground truth bounding boxes. Table III and Table IV illustrate

TABLE III
CLASSIFICATION-LOCALIZATION (CLS-LOC) ERROR ON ILSVRC2014.

Method	top-1 err	top-5 err
GoogLeNet-Backprop [78]	61.31	50.55
GoogLeNet-GAP [11]	56.40	43.00
GoogLeNet-HaS [12]	54.53	–
GoogLeNet-ACoL [14]	53.28	42.58
GoogLeNet-DANet [17]	52.47	41.72
GoogLeNet-mLDFR(Ours)	**49.51**	**38.62**
VGG16-Backprop [78]	61.12	51.46
VGG16-GAP [11]	57.20	45.14
VGG16-ACoL [14]	54.17	40.57
VGG16-ADL [16]	55.08	–
VGG16-I2C [18]	52.59	41.49
VGG16-NLCCAM [77]	49.83	39.31
VGG16-mLDFR(Ours)	**46.96**	**36.41**

TABLE IV
CLASSIFICATION-LOCALIZATION (CLS-LOC) ON CUB200-2011.

Method	top-1 err	top-5 err
GoogLeNet-GAP [11]	58.94	49.34
GoogLeNet-SPG [15]	53.36	42.28
GoogLeNet-DANet [17]	50.55	39.54
GoogLeNet-mLDFR(Ours)	**46.23**	**33.55**
VGG16-GAP [11]	55.85	47.84
VGG16-ADL [16]	52.36	–
VGG16-ACoL [14]	54.08	43.49
VGG16-SPG [15]	51.07	42.15
VGG16-DANet [17]	47.48	38.04
VGG16-NLCCAM [77]	47.60	34.97
VGG16-mLDFR(Ours)	**38.07**	**27.37**

the comparison results. It can be seen that our methods (the networks with mLDFR) significantly outperform all baselines. Especially for the CUB200-2011 dataset, our methods improve the state-of-the art results by a large margin. VGG16-mLDFR obtains 38.07% Cls-Loc Top-1 err on CUB200-2011, which exceeds the state-of-the-art VGG16-DANet [17] by 9.41%. This significant improvement can be attributed to the ability of mLDFR network to simultaneously strengthen the classification ability (discrimination of the features) and location ability (spatial responses of the object) of the network, both of which are important for the Cls-Loc task.

Complexity analysis. Computational efficiency is important for evaluating a WSOL model quality. Here we analyze the efficiency of a model from two aspects: model complexity (the amount of parameters) and computational complexity (FLoating-point OPerations, FLOPs). Three popular backbones for WSOL task are evaluated: VGG16 [68], InceptionV3 [73] and ResNet50 [74]. The number of output categories is set to 1000 (corresponding to ILSVRC). The evaluation results are shown in Table V. It can be seen that, except for original CAM [11] with VGG16 as backbone, our mLDFR framework is more lightweight than most existing WSOL methods in terms of model complexity and computational complexity, even lower than the original CAM [11] with other backbones.

Compared with original CAM [11], for the VGG16 backbone, CAM [11] only add a convolutional layer with the kernel size of 3x3. We not only added such a convolution layer, but also added a set of feature recalibration modules, which generated additional parameters and calculations. For other backbones, [11] did not provide a modification method. The reproduction of subsequent works [53], [54] usually add multiple convolutional layers with the kernel size of 3x3 after

TABLE V
MODEL AND TIME COMPUTATIONAL COMPARISON. THE ASTERISK (*) INDICATES THAT THE RESULT WAS REPRODUCED BY OTHER PAPERS.

Method	Parameters	FLOPs
VGG16-CAM [11]	**20.46M**	**16.31G**
VGG16-ACoL [14]	45.08M	21.34G
VGG16-SPG [15]	29.39M	18.72G
VGG16-DANet [17]	55.12M	25.41G
VGG16-I2C [18]	29.90M	31.46G
VGG16-mLDFR (Ours)	20.82M	16.53G
InceptionV3-CAM* [11], [54]	26.51M	10.79G
InceptionV3-ACoL [14]	44.05M	16.47G
InceptionV3-SPG [15]	38.45M	15.42G
InceptionV3-I2C [18]	27.28M	37.45G
InceptionV3-mLDFR (Ours)	**23.76M**	**8.82G**
ResNet50-CAM* [11], [54]	25.56M	6.25G
ResNet50-ACoL [14]	82.19M	17.75G
ResNet50-SPG [15]	63.31M	15.90G
ResNet50-I2C [18]	54.90M	77.60G
ResNet50-mLDFR (Ours)	**24.78M**	**6.24G**

removing the last downsampling layer and subsequent layers, resulting in a large increase in the amount of parameters and calculations. In our method, we replaced the last downsampling block with a transition layer (as introduced in subsection IV-B) to concatenate before and after, thereby retaining all subsequent convolutional layers. This modification can preserve the learning ability of the network without increasing too many parameters. Compared with other methods [14], [15], [17], [18], the mLDFR framework has obvious advantages in terms of complexity. The main reason is that these methods use additional components or complex structures in the network to find more discriminative parts. For example, ACoL [14] is a parallel two-head architecture, which doubles the amount of parameters and calculations. SPG [15] adds multiple task branches to the network, which brings a lot of additional parameters and calculations.

In summary, our method has advantages in the training phase. However, our model needs to perform three forward propagations in the testing phase to achieve multi-scale class activation mapping (described in Section III-C), so the locating speed is slower than others.

D. Ablation study

Evaluating localization performance with CorLoc. We use ground-truth labels to guide the localization, thereby eliminating the influence caused by classification results. This experiment is called the CorLoc performance test [71]. CorLoc is the percentage of images in which a method correctly localizes an object of the target class according to the Pascal-criterion [79] (IoU>0.5). Table VI shows the comparison of CorLoc on CUB200-2011 and ILSVRC2014, in which our method surpasses all other methods. For ILSVRC2014, we achieve a 0.89% improvement compared to NLCCAM [77]. For CUB200-2011, we increase CorLoc from 67.70% [17] to 75.04%, which is a qualitative leap. This significant progress shows that the mLDFR method is more effective in dealing with WSOL problem in fine-grained dataset. Figure 8 visualizes some results on ILSVRC2014 and CUB200-2011. It can be seen that mLDFR method can generate more complete and compact spatial responses, and can better handle objects of different sizes and multi-instance scenarios.

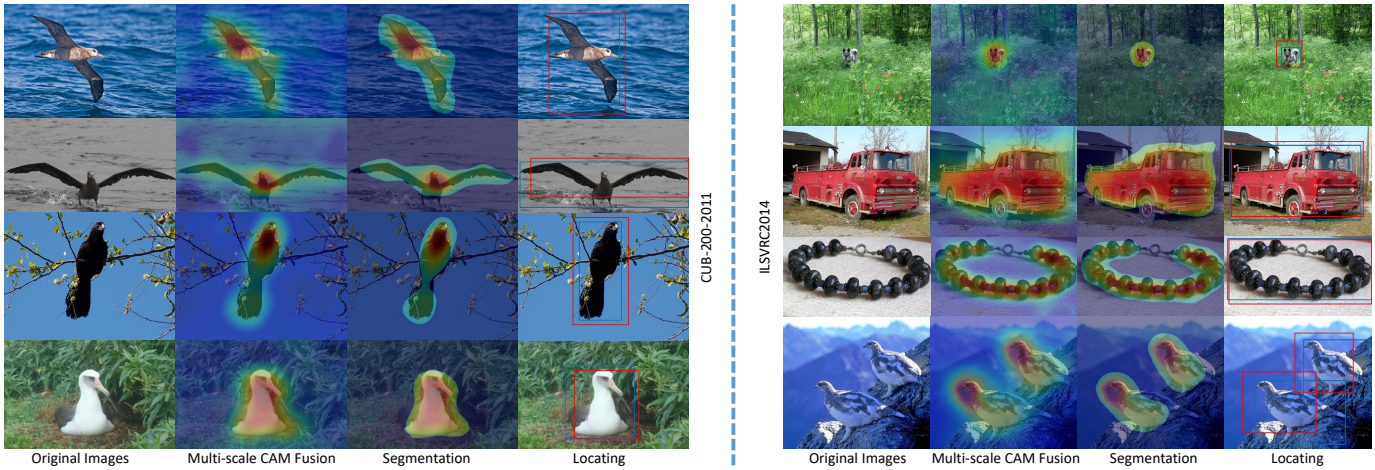


Fig. 8. Localization results on CUB200-2011 (Left) and ILSVRC2014 (Right). Our method can generate more complete and compact spatial responses and handle both single-instance and multi-instance scenes (As shown in the bottom right corner of the figure).

TABLE VI
CORLOC ON CUB200-2011 AND ILSVRC2014.

Method	CUB200-2011	ILSVRC2014
GoogLeNet-CAM [11]	55.10	58.66
GoogLeNet-HaS [12]	–	60.29
GoogLeNet-DANet [17]	67.03	–
GoogLeNet-mLDFR(Ours)	**69.43**	**63.83**
VGG16-CAM [11]	56.00	59.00
VGG16-ACoL [14]	59.30	62.96
VGG16-SPG [15]	58.90	–
VGG16-TSC [80]	65.50	–
VGG16-DANet [17]	67.70	–
VGG16-I2C [18]	–	63.90
VGG16-PSOL [53]	–	64.03
VGG16-NLCCAM [77]	–	65.23
VGG16-mLDFR(Ours)	**75.04**	**66.12**

TABLE VII
MAXBOXACCV2 ON CUB200-2011 AND ILSVRC2014.
'V': VGG16, 'I': INCEPTIONV3, 'R': RESNET50.

	CUB200-2011			ILSVRC2014		
Methods	V	I	R	V	I	R
CAM [11]	63.70	56.70	63.00	60.00	63.40	63.70
HaS [12]	63.70	53.40	64.70	60.60	63.70	63.40
ACoL [14]	57.40	56.20	66.50	57.40	63.70	62.30
SPG [15]	56.30	55.90	60.40	59.90	63.30	63.30
ADL [16]	66.30	58.80	58.40	59.80	61.40	63.70
CutMix [81]	62.30	57.50	62.80	59.40	63.90	63.30
mLDFR (Ours)	**66.83**	**63.09**	**67.28**	**64.65**	**64.08**	**65.19**

Evaluating localization performance with MaxBoxAccV2. The traditional CorLoc protocol [71] requires the box prediction overlaps with at least one of the ground-truth boxes with IoU>0.5, which is a loose standard. Choe et al. [54] suggest averaging CorLoc across different IoU settings {0.3,0.5,0.7} to address diverse demands for localization fineness, and define it as MaxBoxAccV2. This protocol requires the model to have stronger localization stability, that is, the output bbox can more compactly surround the object. Therefore, MaxBoxAccV2 can more clearly reflect the real progress of WSOL models. We used MaxBoxAccV2 to evaluate our method, and the results are shown in Table VII.

As can be seen from the Table VII, with the new evaluation protocol, our method still achieves the best results under the weak supervision setting, indicating that the proposed mLDFR framework can indeed activate the low-discriminative features of the object more adequately than other methods. But it can also be seen from the table that, as [54] said, the performance improvement of WSOL over the years is actually limited.

TABLE VIII
EFFECT OF FR AND MSCAM MODULES.
'F': FR MODULE, 'M': MSCAM MODULE.

	ILSVRC2014		CUB200-2011	
Method	Cls err	CorLoc	Cls err	CorLoc
GoogLeNet	31.07/10.95	56.04	30.74/10.17	25.11
GoogLeNet+F	26.66/8.29	61.45	23.68/5.22	59.12
GoogLeNet+M	31.07/10.95	61.19	30.74/10.17	38.83
GoogLeNet+F+M	26.66/8.29	63.83	23.68/5.22	69.43
VGG16	30.87/10.13	57.78	23.20/6.18	36.04
VGG16+F	24.79/7.32	63.63	19.30/4.19	66.69
VGG16+M	30.87/10.13	61.33	23.20/6.18	42.45
VGG16+F+M	24.79/7.32	66.12	19.30/4.19	75.04

Effect of FR and MsCAM modules. Here we verify the effect of two important components in the mLDFR framework, namely the Feature Recalibration (FR) and the Multi-scale Class Activation Mapping (MsCAM). For VGG16 as backbone, we build four networks according to Section IV-B, namely VGG16, VGG16+FR, VGG16+MsCAM and VGG16+FR+MsCAM. Among them, VGG16 is a basic localization network that does not contain any FR modules and MsCAM. VGG16+FR means that two FR modules (BN + SA) are added to the network, VGG16+MsCAM means using MsCAM in the inference phase of the basic localization network, and VGG16+FR+MsCAM means that both FR and MsCAM modules are added to the network. For GoogLeNet as backbone, we generate four networks in the same way.

It can be seen from the Table VIII that both FR and MsCAM modules can effectively improve the localization performance of the model. On the ILSVRC dataset, FR and MsCAM modules can stably improve the localization performance of the model. On the CUB200 dataset, the performance improvements brought by the FR modules is more significant. This is because in the training process of the fine-grained dataset, a large number of similar features between object categories are discarded by the model, and the recalibration modules can effectively activate these discarded features.

TABLE IX
EFFECT OF DIFFERENT FR MODULES. 'S': SA MODULE, 'B': BN MODULE.

	ILSVRC2014		CUB200-2011	
Method	Cls err	CorLoc	Cls err	CorLoc
GoogLeNet	31.07/10.95	61.19	30.74/10.17	38.83
GoogLeNet+S	28.49/9.61	63.07	28.58/8.18	44.55
GoogLeNet+B	27.15/8.64	63.26	23.73/5.42	68.03
GoogLeNet+B+S	26.66/8.29	63.83	23.68/5.22	69.43
VGG16	30.87/10.13	61.33	23.20/6.18	42.45
VGG16+S	25.91/7.87	63.35	21.92/5.51	47.21
VGG16+B	26.31/8.29	63.49	19.65/4.91	71.82
VGG16+B+S	24.79/7.32	66.12	19.30/4.19	75.04
ResNet50X	27.22/9.14	61.67	40.89/20.76	10.43
ResNet50X+S	24.74/8.06	63.84	23.69/6.26	37.50
ResNet50X+B	23.24/7.05	63.81	22.48/5.66	72.84
ResNet50X+B+S	23.17/6.85	67.55	21.15/4.57	78.67

TABLE X
EFFECT OF DIFFERENT MSCAM SETTINGS. ALL SETTINGS EXCEPT THE NUMBER OF IMAGE PYRAMID LAYERS ARE CONSISTENT.

Backbone	Scales	ILSVRC2014	CUB200-2011
GoogLeNet	1	61.45	59.12
	1,1.5	63.03	66.14
	1,1.5,2	63.83	69.43
	1,1.5,2,2.5	64.05	70.54
VGG16	1	63.63	66.69
	1,1.5	65.56	72.60
	1,1.5,2	66.12	75.04
	1,1.5,2,2.5	66.53	75.70

Effect of different feature recalibration (FR) modules. Here we investigate how much the two recalibration modules contribute to improve the classification and localization performance. For VGG16 as backbone, we build four networks according to Section IV-B, namely VGG16, VGG16+SA, VGG16+BN and VGG16+BN+SA. Among them, VGG16 is a basic localization network that does not contain any recalibration modules. VGG16+SA means that only the proposed SA module is added to the network, VGG16+BN means that only BN module is added to the network, and VGG16+BN+SA means that both BN module and SA module are added to the network. For GoogLeNet as backbone, we generate four networks in the same way. In order to further verify the effectiveness of our method, we also chose ResNet50 [74] as the backbone. Since there is no widely used version of the ResNet50 backbone that does not include the BN module, we modified the original ResNet50 [74] and removed all BN modules, and used the modified network for testing. We named the modified network ResNet50X. We apply MsCAM in all experiments. Table IX show the comparisons. On the ILSVRC2014, BN and SA modules steadily improve the classification and localization performance for all backbones. On the CUB200-2011, BN plays a more important role in improving the classification and localization performance of the network. Especially in localization task, with VGG16 as backbone, CorLoc is promoted from 42.45% to 71.82%, and with ResNet50X as backbone, CorLoc is promoted from 10.43% to 72.84%. The impact of the SA module is relatively small. It is probably because that the CUB200-2011 is composed of bird sub-categories, resulting in a higher channel correlation between instances in a batch. So the cross-instance global spatial information aggregation of BN module can better capture the common features of birds.

TABLE XI
EFFECT OF MULTI-SCALE TRAINING. 'W' AND 'W/O' RESPECTIVELY INDICATE WHETHER TO USE MULTI-SCALE IN TRAINING PHASE.

	ILSVRC2014		CUB200-2011	
Backbone	Cls err	CorLoc	Cls err	CorLoc
VGG16 (w/o)	25.30/7.78	66.09	21.00/5.90	67.29
VGG16 (w/)	24.79/7.32	66.12	19.30/4.19	75.04
GoogLeNet (w/o)	26.46/8.31	63.05	32.27/9.25	57.85
GoogLeNet (w/)	26.66/8.29	63.83	23.68/5.22	69.43

TABLE XII
EFFECT OF BATCH SIZE. '–' MEANS THAT THE MODEL CANNOT CONVERGE.

	ILSVRC2014		**CUB200-2011**	
Batchsize	**Cls err**	**CorLoc**	**Cls err**	**CorLoc**
16	–/–	–	–/–	–
32	27.01/8.54	66.05	28.87/8.87	62.98
64	25.63/7.85	66.71	20.07/4.76	73.99
128	25.07/7.78	66.69	19.46/4.35	75.17
256	24.79/7.32	66.12	19.30/4.19	75.04

Effect of different MsCAM settings. To verify the effect of MsCAM in WSOL, we test different MsCAM settings after building the network. The results are listed in Table X. It can be observed that the more the number of image pyramid layers, the better the locating effect. This demonstrates that through the MsCAM, neurons can capture more different and complementary parts of the object at different scales. However, as the number of scales increases, the improvement becomes slower, and the calculation time increases. Therefore, in order to balance performance and speed, we recommend using a three-layer image pyramid. However, in complex scenes, such as containing particularly small or large instance, and multiple instances, a pyramid with more layers can be used.

Effect of multi-scale training. Multi-scale is essential to mLDFR method. Here we investigate the impact of multi-scale training on the classification and localization performance of the model. We select VGG16 as backbone and verify on ILSVRC2014 and CUB200-2011 datasets. The results are shown in Table XI. 'w' and 'w/o' respectively indicate whether to use multi-scale augmentation in the training phase.

It can be seen that multi-scale training has almost no impact on the performance of the ILSVRC dataset, but has a greater impact on the CUB200 dataset, especially for localization task. The reason is as follows: 1) For ILSVRC, there are a total of 1281167 images for training. The number of images for each category ranges from 732 to 1300. The objects in ILSVRC dataset have sufficient diversity in scale. Therefore, in this case, even without multi-scale augmentation, the trained model is still scale-invariant, that is, it can find different discriminative features on objects of different scales. 2) For CUB200, there are only 5994 images for training, averaging less than 30 samples per category. The lack of data leads to insufficient diversity of the dataset. In this case, multi-scale augmentation can effectively improve the scale diversity of the dataset and strengthen the robustness of the model to scale change, thereby achieving better localization performance.

Effect of batch size The BN module uses specific channel information of all instances in a batch to calculate statistic variables [24]. Therefore, for BN module, in theory, the larger the number of instances in the batch, the better it is to calculate effective statistics. If the batch size is too small, there will be

TABLE XIII
EFFECT OF DIFFERENT BACKBONES. THE ASTERISK (*) INDICATES THAT THE RESULT WAS REPRODUCED BY OTHER PAPERS.

	ILSVRC2014					CUB200-2011				
Method	Cls err		Cls-Loc err		CorLoc	Cls err		Cls-Loc err		CorLoc
InceptionV3-CAM* [11], [53]	26.70	8.20	53.71	41.81	62.68	–	–	56.33	46.47	–
InceptionV3-SPG [15]	30.30	9.90	51.40	40.00	64.69	–	–	53.36	42.28	–
InceptionV3-ADL [16]	27.17	–	51.29	–	–	25.45	–	46.96	–	–
InceptionV3-I2C [18]	26.70	8.40	46.89	**35.87**	**68.50**	–	–	44.01	31.66	72.60
InceptionV3-mLDFR	**21.78**	**5.72**	**45.87**	36.36	64.94	**20.42**	**5.04**	**39.24**	**28.39**	**74.41**
ResNet50-CAM* [11], [53]	–	–	61.01	50.53	51.86	–	–	70.42	62.75	–
ResNet50-SE-ADL [16]	24.15	–	51.47	–	–	**19.66**	–	37.71	–	–
ResNet50-I2C [18]	23.30	6.90	**45.17**	**35.40**	**68.50**	–	–	–	–	–
ResNet50-mLDFR	**23.17**	**6.85**	45.96	35.78	67.55	21.15	**4.57**	**36.41**	**24.11**	**78.67**
DenseNet161-CAM* [11], [53]	–	–	60.39	49.60	52.54	–	–	70.19	60.15	–
DenseNet161-mLDFR	**21.55**	**5.58**	**43.37**	**33.58**	**69.06**	**20.59**	**4.74**	**35.41**	**23.11**	**79.91**
MobileNetV1-CAM* [11], [16]	31.62	–	58.34	–	–	28.06	–	56.30	–	–
MobileNetV1-HaS* [12], [16]	32.52	–	58.13	–	–	33.36	–	55.33	–	–
MobileNetV1-ADL [16]	32.23	–	56.99	–	–	29.57	–	52.26	–	–
MobileNetV2-mLDFR	**29.89**	**10.22**	**52.35**	**41.01**	**63.16**	**26.14**	**7.49**	**43.25**	**30.46**	**73.48**

a lot of statistical noises. To verify the effect of the number of instances in a batch on the performance of the model, we chose VGG16 as backbone to conduct comparative experiments, in which all settings except the batch size are consistent. It can be seen from Table XII that: a) As the batch size increases, the performance of the model tends to saturate, indicating that too large batch size will not bring more benefits. When the batch size is reduced to a certain level (e.g. 16), the training cannot converge. This shows that sufficient instances are important for calculating effective statistics in the BN module. b) When the batch size is reduced from 64 to 32, the performance decreases a lot. Especially for CUB200, the classification and localization performance show a significant decline. This shows that for fine-grained datasets, global spatial information aggregation across instances is more important.

Effect of different backbones. The reason why we use VGG16 and GoogLeNet is that these two backbones do not contain the BN modules, which allows us to verify the feature recalibration effect of the BN module. In order to verify the compatibility of the mLDFR framework to various types of backbones, we select four new backbones for verification. The details are as follows: *InceptionV3 [73]*, published in 2016, is an upgraded version of GoogLeNet [72], which further improves the inception structure; *ResNet50 [74]*, published in 2016, is the first network to introduce residual modules; *DenseNet161 [75]*, published in 2017, uses dense connections between layers on the basis of ResNet [74]; *MobileNetV2 [76]*, published in 2018, is a lightweight network structure for low-power environments. The detailed modification methods are described in Section IV-B. The results are shown in Tables XIII. We compared with other methods according to the backbone. Since there are fewer papers based on these backbones, we also collected similar backbones when conducting the experiment comparison (e.g. ResNet50-SE [20], MobileNetV1 [82]). From Table XIII, it can be seen that:

1) The mLDFR achieves the stable performance improvements on various types of backbones, which proves that the proposed mLDFR framework has good compatibility. mLDFR surpasses all other methods in Top1/Top5 classification error, showing that our method can maintain the classification ability while improving the localization ability of the model.

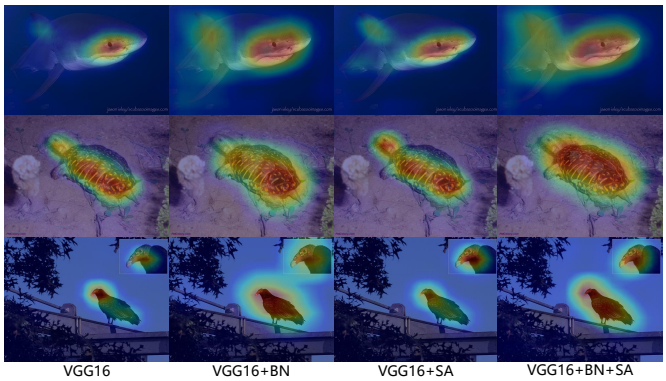


Fig. 9. The effect of different FR modules on CAM.

2) On CUB200, the mLDFR framework achieves the best classification and localization results on all backbones. Especially with DenseNet161 as backbone, DenseNet161-mLDFR achieves a CorLoc of 79.91%, which greatly improves the localization performance. This is probably because the dense connection of DenseNet161 backbone enables the localization layer to fully perceive the features of each level from the bottom to the top, which is beneficial for localization task.

3) On ILSVRC, DenseNet161-mLDFR also achieves the best localization performance. mLDFR based on ResNet50 achieves a CorLoc of 67.55%, which is still competitive (0.95% lower than I2C [18]). But when using InceptionV3 as backbone, the CorLoc of our method is 64.94%, which is 3.56% lower than I2C [18], In addition, for GoogLeNet[72] and InceptionV3 [73] backbones based on the inception structure, the improvement in localization performance is relatively small. Although the more advanced inception structure (InceptionV3) obtains almost the best classification performance, its localization performance is still not as good as a simple full convolution structure (VGG16). This shows that the mLDFR framework does not fully adapt to the inception structure.

Visualization of the effect of different FR modules on CAM. In order to visually compare the influence of different recalibration modules on the spatial response of the object, we visualize the CAM of four networks with VGG16 as the backbone, namely VGG16, VGG16+SA, VGG16+BN and VGG16+BN+SA. It can be seen that the improvement of

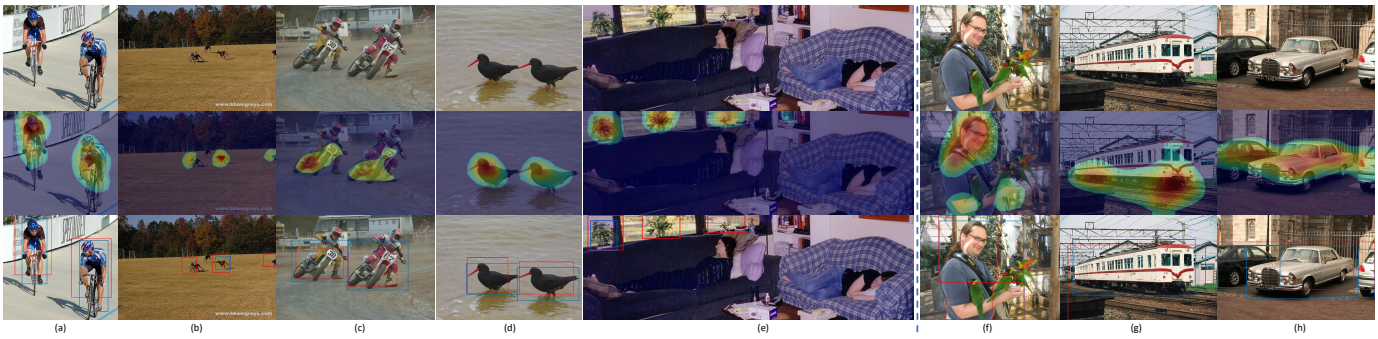


Fig. 10. Localization results in complex scenes. Top: original image, Middle: segmentation, Bottom: localization boxes in red and groundtruth boxes in blue. (a)-(e) shows examples of multi-instance localization of different sizes. (f)-(h) shows the limitations of the our method in multi-instance localization task.

BN and SA modules to the object's spatial response is very significant, indicating that these two modules can effectively activate the object's low discriminative features. And the two recalibration modules can be integrated to further improve object responses. At the same time, compared to BN, the SA module is more helpful for discovering new object parts, as shown in the second and third rows of Figure 9.

E. Discussion

Multi-instance Detection. The existing AL-based methods [19], [12], [14], [15], [16] are applied to datasets of simple scenes, such as ILSVRC and CUB200, which contain only one instance in most images. Our mLDFR framework avoids fusion operations for the response regions, and captures more objects through feature recalibration and multi-scale activation. Therefore, it is able to handle the multi-instance localization in complex scenes. We verify this point on VOC2007 [79].

PASCAL VOC2007 [79] contains 20 object categories, consisting of 2.5k training images, 2.5k validation images, and 5k test images, with an average of 1.4 categories and 2.5 instances per image, and 44.5% of the images contain 1 instance. Considering that the object shape in VOC2007 is complex, we build an image pyramid with a larger span [0.5,1.0,1.5,2.0,2.5] to capture more object features. We generate bounding boxes on each connected region, where each bounding box corresponds to an object instance.

In the VOC2007 test set, according to the PASCAL criterion in object detection, the VGG16-mLDFR obtains the mAP of 15.51%, 31.81%, 52.76% for region overlap thresholds 0.5, 0.3, 0.1, respectively. Figure 10 visualizes some results. The performance is not high because it is hard to solve some inherent limitations of CAM-based paradigm, as follows.

1) Some features of the object category are too low-discriminative, which makes them unable to respond on the CAM. For example, for the human category, head and hands have a higher discrimination, while the clothes are less discriminative (As shown in Figure 10-(f)).

2) As a non-linear statistical model, convolutional neural network is difficult to separate co-occurrence patterns when data is limited and lack of priors (such as object-level annotations). In other words, the network regards the co-occurrence pattern as a whole, which leads to excessive localization under weakly supervision (As shown in Figure 10-(g)).

TABLE XIV
CORLOC ON VOC2007 TRAINVAL. THE UNDERLINED ACCURACIES INDICATE THAT THE SCORES ARE RETESTED BY US.

Method	VOC2007
MIL-based	
WSDDN [40]	58.00
OICR [41]	60.60
PCL [44]	62.70
WSRPN [43]	63.80
CAP+SRN [45]	66.70
OIM [49]	67.20
SDCN [47]	68.60
CAM-based	
VGG16-SP [13]	40.61
VGG16-mLDFR(Ours)	47.91

3) CNN focuses on semantics rather than number of object instances. When the instances are very close or overlap, the CAM-based method is difficult to distinguish the boundaries between the instances (As shown in Figure 10-(h)).

Although the effect of multi-instance detection is still poor, our method still shows a certain potential, as shown in Figure 10 (a)-(e). Moreover, because the CAM-based method completes the object detection while image classifying, it has obvious advantages in detection speed.

Pseudo Object-Level Annotation Mining in Complex Scene. As far as we know, in complex scene (such as VOC [79] and MSCOCO [83]), almost all works use the MIL-based paradigm [39], [40], [41], [42], [43], [44], [45], [46], [47], [49] for pseudo object-label mining, while few works use the CAM-based paradigm [13]. CorLoc is usually used to measure the performance of the model in pseudo-label mining tasks. Table XIV shows the comparison results of CAM-based methods and MIL-based methods on VOC2007 [79] trainval. It can be seen that the performance of CAM-based method in pseudo-label mining task is significantly lower than that of MIL-based method. The reason mainly lies in:

1) CAM-based methods look for the objects directly in the image, which is a huge search space. And the MIL-based methods actually face a much smaller and higher quality search space. For example, for VOC2007 [79], SelectiveSearch [50] can recall more than 98% of bounding boxes of groundtruth in the generated 2000 regional proposals according to the Pascal-criterion [79] (IoU>0.5).

2) The CAM-based methods are difficult to deal with: (a) co-occurrence patterns; (b) objects are too close or overlapped. As we analyzed in sub-section IV-E. Especially for the AL

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIP.2021.3091833, IEEE Transactions on Image Processing

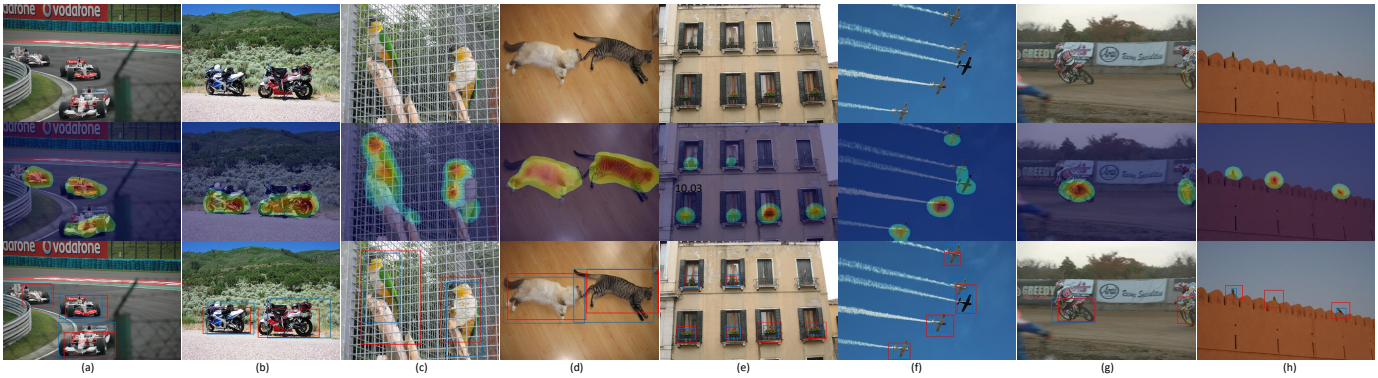


Fig. 11. Pseudo object-level annotation mining. Our method can discovers multiple instances and even find instances that are not labeled in the groundtruth.

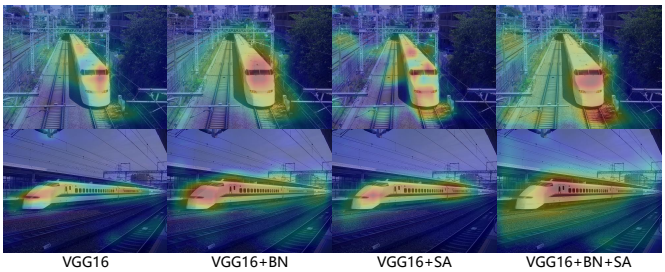


Fig. 12. The activation map of train. Since the train and the rail are highly correlated in dataset, the recalibration module also strengthens their features.

framework, it is inherently unable to perform multi-instance localization task (as shown in the bottom row of Figure 1).

However, CAM-based method still shows the potential and advantages of pseudo-label mining in complex scenes. As shown in Figure 11, our method can simultaneously mine multiple category instances and avoid false-negative samples, which is difficult to achieve under the MIL-based framework. In the complex scenes, where the category instances are not obvious (such as Figure 11-(g) and (h)), our method even finds category instances that are not labeled in the groundtruth. This undoubtedly greatly saves labor costs in object detection task.

How the FR module affects the co-occurrence patterns. Although the recalibration module can effectively improve the localization performance of the model, it can cause serious locating errors when dealing with co-occurrence patterns. As shown in Figure 12, The co-occurrence patterns refer to a situation in which a foreground category is statistically highly correlated with a certain background category in the dataset. As a non-linear statistical modeling tool, convolutional neural network is difficult to separate co-occurrence patterns when data is limited and lack of priors. In other words, in the learning process, the network regards the co-occurrence pattern as a whole, such as train and rail. Therefore, for co-occurrence patterns, any recalibration module theoretically will simultaneously strengthen the feature representation of the foreground and the background, including SA and BN.

How the CAM-based method solves inherent limitations. The classification task aims to find more discriminative and higher semantic features, while the localization task needs to find the complete object features, and at the same time need to pay attention to the low-level features such as edges and contours. These two tasks are essentially contradictory. Therefore, in the absence of sufficient data and prior knowledge, the inherent limitations (i.e. low-discriminative object regions, co-concurrence patterns, and close/overlapped objects) of CAM-based framework discussed in sections IV-E and IV-E are difficult to resolve. But some possible solutions are as follows.

1) For low-discriminative parts. Since CNN is a statistical model, it is difficult for the network to model the relationship between the low-discriminative parts and the labels in an open feature space when the data is limited. Therefore, a data-augmentation based solution is feasible, which can statistically enhance the correlation between the low-discriminative parts and the category label.

2) For co-concurrence patterns. The essence of the co-occurrence patterns is that the posterior probability of some foreground parts is less than the posterior probability of the background parts. Therefore, the possible solution is still a data-augmentation strategy. We can add positive samples containing low-discriminative parts and negative samples containing only background. In addition, low-level features (such as color, texture, edge) are also helpful for distinguishing the foreground and the background.

3) For close/overlapped objects. The classification network does not care about the number of object instances, so the CAM based on the classification network is difficult to locate the boundaries between different object instances. In the post-processing stage, low-level features (such as color, texture, etc.) can be used to capture the contour of the object instance or locate the boundaries between object instances in the image, and combine the processing results with the score map to separate multiple object instances.

V. Conclusion

In this paper, we proposed a new WSOL framework under CAM-based paradigm, called Multi-scale Low-Discriminative Feature Reactivation (mLDFR). The framework consists of two core components, i.e. bottom-up continuous feature maps recalibration and multi-scale object category mapping. Compared with the Adversarial Learning based methods, our method solves the conflict between classification and localization performance and can perform multi-instance localization. Experiments on widely used datasets have demonstrated the insights revealed in this paper as well as the effectiveness of the proposed approach. In the future, we will further study

the feature communication method between various object parts to activate more object regions. Moreover, we will study how to obtain feature maps with higher resolution while remaining classification performance. This will help achieve more accurate object localization and segmentation.

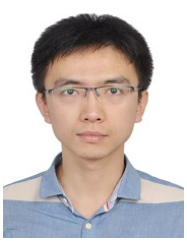
ACKNOWLEDGMENT

This work is supported by the National Key R&D Plan (No. 2018YFC0823003, 2017YFB1002801), Beijing Natural Science Foundation (No. L182058), the Natural Science Foundation of China (No. 61876100, 61972397), Shandong Provincial Science and Technology Support Program of Youth Innovation Team in Colleges (No. 2019KJN041).

REFERENCES

[1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014, pp. 580–587.
[2] R. Girshick, "Fast r-cnn," in *ICCV*, 2015, pp. 1440–1448.
[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *NeurIPS*, 2015.
[4] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *ECCV*, 2016.
[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016, pp. 779–788.
[6] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *CVPR*, 2017, pp. 7263–7271.
[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *ICCV*, 2017, pp. 2961–2969.
[8] H. O. Song, Y. J. Lee, S. Jegelka, and T. Darrell, "Weakly-supervised discovery of visual pattern configurations," in *NeurIPS*, 2014.
[9] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Is object localization for free?-weakly-supervised learning with convolutional neural networks," in *CVPR*, 2015, pp. 685–694.
[10] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene cnns," in *International Conference on Learning Representations*, 2015.
[11] ——, "Learning deep features for discriminative localization," in *CVPR*, 2016, pp. 2921–2929.
[12] K. K. Singh and Y. J. Lee, "Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization," in *ICCV*, 2017, pp. 3544–3553.
[13] Y. Zhu, Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao, "Soft proposal networks for weakly supervised object localization," in *ICCV*, 2017, pp. 1841–1850.
[14] X. Zhang, Y. Wei, J. Feng, Y. Yang, and T. S. Huang, "Adversarial complementary learning for weakly supervised object localization," in *CVPR*, 2018.
[15] X. Zhang, Y. Wei, G. Kang, Y. Yang, and T. Huang, "Self-produced guidance for weakly-supervised object localization," in *ECCV*, 2018.
[16] J. Choe and H. Shim, "Attention-based dropout layer for weakly supervised object localization," in *CVPR*, 2019.
[17] H. Xue, C. Liu, F. Wan, J. Jiao, X. Ji, and Q. Ye, "Danet: Divergent activation for weakly supervised object localization," in *ICCV*, 2019.
[18] X. Zhang, Y. Wei, and Y. Yang, "Inter-image communication for weakly supervised localization," in *ECCV*, 2020.
[19] D. Kim, D. Cho, D. Yoo, and I. So Kweon, "Two-phase learning for weakly supervised object localization," in *ICCV*, 2017, pp. 3534–3543.
[20] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *CVPR*, 2018, pp. 7132–7141.
[21] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, "Gather-excite: Exploiting feature context in convolutional neural networks," in *NeurIPS*, 2018, pp. 9401–9411.
[22] J. Park, S. Woo, J.-Y. Lee, and I. S. Kweon, "Bam: Bottleneck attention module," in *BMVC*, 2018.
[23] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *ECCV*, 2018, pp. 3–19.
[24] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.
[25] B. Zhang, J. Xiao, Y. Wei, M. Sun, and K. Huang, "Reliability does matter: An end-to-end weakly supervised semantic segmentation approach," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.
[26] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *CVPR*, 2001.
[27] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *CVPR*, 2010.
[28] L. Ladickỳ, C. Russell, P. Kohli, and P. H. Torr, "Associative hierarchical crfs for object class image segmentation," in *ICCV*, 2009, pp. 739–746.
[29] J. I. Olszewska, "Designing transparent and autonomous intelligent vision systems." in *ICAART (2)*, 2019, pp. 850–856.
[30] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *IJCV*, pp. 321–331, 1988.
[31] J. I. Olszewska, "Active contour based optical character recognition for automated scene understanding," *Neurocomputing*, pp. 65–71, 2015.
[32] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, pp. 834–848, 2017.
[33] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, "Context encoding for semantic segmentation," in *CVPR*, 2018, pp. 7151–7160.
[34] X. J. Zhu, "Semi-supervised learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2005.
[35] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009.
[36] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, pp. 44–53, 2018.
[37] J. A. Olvera-López, J. A. Carrasco-Ochoa, J. F. Martínez-Trinidad, and J. Kittler, "A review of instance selection methods," *Artificial Intelligence Review*, pp. 133–143, 2010.
[38] J. L. Carbonera and J. I. Olszewska, "Local-set based-on instance selection approach for autonomous object modelling," *International Journal of Advanced Computer Science and Applications*, 2019.
[39] A. J. Bency, H. Kwon, H. Lee, S. Karthikeyan, and B. Manjunath, "Weakly supervised localization using deep feature maps," in *ECCV*, 2016, pp. 714–731.
[40] H. Bilen and A. Vedaldi, "Weakly supervised deep detection networks," in *CVPR*, 2016.
[41] P. Tang, X. Wang, X. Bai, and W. Liu, "Multiple instance detection network with online instance classifier refinement," in *CVPR*, 2017.
[42] Y. Wei, Z. Shen, B. Cheng, H. Shi, J. Xiong, J. Feng, and T. Huang, "Ts2c: Tight box mining with surrounding segmentation context for weakly supervised object detection," in *ECCV*, 2018, pp. 434–450.
[43] P. Tang, X. Wang, A. Wang, Y. Yan, W. Liu, J. Huang, and A. Yuille, "Weakly supervised region proposal network and object detection," in *ECCV*, 2018, pp. 352–368.
[44] P. Tang, X. Wang, S. Bai, W. Shen, X. Bai, W. Liu, and A. Yuille, "Pcl: Proposal cluster learning for weakly supervised object detection." *IEEE transactions on pattern analysis and machine intelligence*, 2018.
[45] S. Kosugi, T. Yamasaki, and K. Aizawa, "Object-aware instance labeling for weakly supervised object detection," in *ICCV*, 2019, pp. 6064–6072.
[46] K. Yang, D. Li, and Y. Dou, "Towards precise end-to-end weakly supervised object detection network," in *ICCV*, 2019, pp. 8372–8381.
[47] X. Li, M. Kan, S. Shan, and X. Chen, "Weakly supervised object detection with segmentation collaboration," in *ICCV*, 2019.
[48] Y. Shen, R. Ji, K. Yang, C. Deng, and C. Wang, "Category-aware spatial constraint for weakly supervised detection," *TIP*, 2020.
[49] C. Lin, S. Wang, D. Xu, Y. Lu, and W. Zhang, "Object instance mining for weakly supervised object detection," *arXiv:2002.01087*, 2020.
[50] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *IJCV*, pp. 154–171, 2013.
[51] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014, pp. 391–405.
[52] J. Zhang, S. A. Bargal, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-down neural attention by excitation backprop," *IJCV*, 2018.
[53] C.-L. Zhang, Y.-H. Cao, and J. Wu, "Rethinking the route towards weakly supervised object localization," in *CVPR*, 2020.
[54] J. Choe, S. J. Oh, S. Lee, S. Chun, Z. Akata, and H. Shim, "Evaluating weakly supervised object localization methods right," in *CVPR*, 2020.
[55] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *NeurIPS*, 2015, pp. 2017–2025.
[56] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *CVPR*, 2017, pp. 3156–3164.
[57] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and Ł. Kaiser, "Attention is all you need," in *NeurIPS*, 2017.
[58] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *CVPR*, 2018, pp. 7794–7803.

[59] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint:1607.06450*, 2016.
[60] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv:1607.08022*, 2016.
[61] Y. Wu and K. He, "Group normalization," in *ECCV*, 2018, pp. 3–19.
[62] P. Luo, J. Ren, Z. Peng, R. Zhang, and J. Li, "Differentiable learning-to-normalize via switchable normalization," *arXiv:1806.10779*, 2018.
[63] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," in *ICLR*, 2017.
[64] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *ICCV*, 2017, pp. 1510–1519.
[65] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," in *CVPR*, 2017, pp. 4105–4113.
[66] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014, pp. 818–833.
[67] S. Pereira, A. Pinto, J. Amorim, A. Ribeiro, V. Alves, and C. A. Silva, "Adaptive feature recombination and recalibration for semantic segmentation with fully convolutional networks," *IEEE transactions on medical imaging*, pp. 2914–2925, 2019.
[68] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint:1409.1556*, 2014.
[69] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, pp. 211–252, 2015.
[70] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.
[71] T. Deselaers, B. Alexe, and V. Ferrari, "Weakly supervised localization and learning with generic knowledge," *IJCV*, pp. 275–293, 2012.
[72] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015, pp. 1–9.
[73] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.
[74] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.
[75] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017, pp. 2261–2269.
[76] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *CVPR*, 2018.
[77] S. Yang, Y. Kim, Y. Kim, and C. Kim, "Combinational class activation maps for weakly supervised object localization," in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020.
[78] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint:1312.6034*, 2013.
[79] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *IJCV*, pp. 303–338, 2010.
[80] X. He and Y. Peng, "Weakly supervised learning of part selection model with spatial constraints for fine-grained image classification," in *Thirty-first AAAI conference on artificial intelligence*, 2017.
[81] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *ICCV*, 2019.
[82] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv:1704.04861*, 2017.
[83] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014, pp. 740–755.



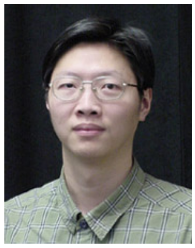
Bo Wang is currently an research assistant with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China. He is also pursuing the Ph.D. degree in the School of Software and Microelectronics, Peking University. His research interests include machine learning, image and video understanding, and multimedia content security.

Chunfeng Yuan received the Ph.D. degree from National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences(CASIA), Beijing, China, in 2010. She is currently an associate professor with CASIA. Her research interests and publications range from statistics to computer vision, including sparse representation, motion analysis, action recognition, and event detection.

Bing Li received the Ph.D. degree from the Department of Computer Science and Engineering, Beijing Jiaotong University, Beijing, China, in 2009. From 2009 to 2011, he worked as a postdoctoral research fellow with National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences(CASIA), Beijing. He is currently a professor with CASIA. His current research interests include computer vision, color constancy, visual saliency detection, multi-instance learning, and data mining.

Xinmiao Ding received the Ph.D. degree in mechanical, electronic, and information engineering from the China University of Mining and Technology, Beijing, China, in 2013. From March 2015 to March 2016, she is a Visiting Scholar with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. Her main research interests include image and video analysis and understanding, machine learning, and internet security.

Zeya Li received the M.S. degree from Shanghai Jiaotong University, Shanghai, China, in 2016. She is currently an engineer with the Beijing Institude of Tracking and Telecommunications Technology. Her research interests include satellite system design, spacecraft design, engineering automati.



Ying Wu received the B.S. degree from the Huazhong University of Science and Technology, Wuhan, China, the M.S. degree from Tsinghua University, Beijing, China, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign (UIUC), Urbana, Illinois, in 1994, 1997, and 2001, respectively. In 2001, he joined the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL, USA, as an Assistant Professor. He was promoted as an Associate Professor in 2007 and a Full Professor in 2012. His current research interests include computer vision, image and video analysis, pattern recognition and machine learning. He is a Fellow of the IEEE, for his "fundamental contributions to visual motion analysis and visual pattern discovery in computer vision".

Weiming Hu received the Ph.D. degree from the Department of Computer Science and Engineering, Zhejiang University, Zhejiang, China, in 1998. From 1998 to 2000, he was a postdoctoral research fellow with the Institute of Computer Science and Technology, Peking University, Beijing. He is currently a professor with the Institute of Automation, Chinese Academy of Sciences(CASIA), Beijing. His research interests are visual motion analysis, recognition of web objectionable information, and network intrusion detection.