MinMaxCAM: Improving object coverage for CAM-based Weakly Supervised Object Localization

Kaili Wang[1][2]
kaili.wang@esat.kuleuven.be
Jose Oramas[2]
Jose.Oramas@uantwerpen.be
Tinne Tuytelaars[1]
tinne.tuytelaars@esat.kuleuven.be

[1] KU Leuven, ESAT-PSI
Leuven, Belgium

[2] University of Antwerp, imec-IDLab
Antwerp, Belgium

Abstract

One of the most common problems of weakly supervised object localization is that of inaccurate object coverage. In the context of state-of-the-art methods based on Class Activation Mapping, this is caused either by localization maps which focus, exclusively, on the most discriminative region of the objects of interest, or by activations occurring in background regions. To address these two problems, we propose two representation regularization mechanisms: *Full Region Regularization* which tries to maximize the coverage of the localization map inside the object region, and *Common Region Regularization* which minimizes the activations occurring in background regions. We evaluate the two regularizations on the ImageNet, CUB-200-2011 and OpenImages-segmentation datasets, and show that the proposed regularizations tackle both problems, outperforming the state-of-the-art by a significant margin.

1 Introduction

Learning how to localize objects in images without relying on data paired with expensive location-specific annotations is a highly desirable capability. Therefore, it is no surprise that this task, usually referred to as Weakly-supervised Object Localization (WSOL), has received increased attention in recent years.

One of the most used methods for this task is based on Class Activation Map (CAM) [23], see [3, 15, 18, 20, 21, 22]. In these works, it has been noticed that the localization map generated by CAM focuses on the most discriminative region of the image. The reason is simple: the backbone is trained for classification since there is no access to the coordinates of the object, so it learns the discriminative features for each class. As a result, the object coverage is under-estimated.

Existing efforts to address this problem follow one of three common strategies. They iteratively occlude/replace relevant regions of the input in order to force the model to learn a more complete set of features, enabling localization [15, 18], or rely on additional networks to assist with the localization task [20, 21]. Alternatively, a more simple strategy is to update the representation learned by the convolutional layers of the model [3, 22].

For CAM-based methods, the localization map is generated by a linear combination of the feature maps of the last convolutional layer and then rescaled to the image's size. If the size of the feature map is too small, say 7×7 while the input size is 224×224, the resized localization map will have poor precision. In order to increase the size while still using the original pre-trained weights of the backbone, a common strategy is to change the stride in some convolutional blocks [3, 5, 22]. However, we observe that the localization map generated in this way can activate a lot on the background, i.e. the object coverage is over-estimated (see Fig. 2). To solve it, activations on the background should be suppressed.

Then, the research question is: *how to actively control the activation distribution on the localization map, maximizing or minimizing the spatial coverage (mass) of the activations as needed?* Aiming at answering this question, we propose *MinMaxCAM*, a method that iteratively i) learns a classification task to provide coarse feature maps (Stage I) and ii) regularizes (part of) the model (i.e. the final linear layer, noted as fc), which learns how to re-weight the coarse feature maps so that it is capable of shifting the mass of their internal activations (Stage II), see Fig. 1. This not only enables accurate object localization but is relatively stable to train and does not need additional networks. In particular, we design two regularizations, *Common Region Regularization (CRR)* and *Full Region Regularization (FRR)*, that can serve as objective functions for the model to optimize fc after the global average pooling (GAP) operation characteristic of CAM. *CRR* is based on the fact that multiple images from the same class share a very similar representation for the common object, which we call "object-specific representation". During the training, the coarse localization map obtained from Stage I via CAM is used to extract the object-specific representation. When minimizing *CRR*, we only optimize the final linear layer, whose weights are used by CAM to combine the feature maps, making the localization map more accurate. Worth noting, intra-class differences can reduce the common region to only a very small part of the object. The same can happen due to failed localization of the most discriminative region. To tackle these situations, *FRR* is proposed. It stimulates covering a larger part of the object, again only optimizing the final linear layer. Stage I and II are optimized every mini-batch, where optimizing Stage I guarantees that the coarse localization map is object-centered.

MinMaxCAM has a number of advantages: i) It is light-weight: it only relies on a standard classification model; no extra network is needed. It saves computation resources and is relatively simple to train. ii) The proposed method produces more precise or tighter bounding boxes, addressing the problem of over- and under-estimating the object with a single model. iii) Despite its simplicity, the proposed method is capable of setting a new state-of-the-art performance on the ImageNet, CUB-200-2011 and OpenImages-segmentation datasets, outperforming existing methods by a significant margin.

2 Related Work

Most existing works related to WSOL [3, 15, 17, 18, 20, 21, 22] are based on Class Activation Mapping (CAM) [23]. They address the WSOL task, indirectly, by solving the problem that the generated localization map only focuses on the most discriminative regions of the image. These methods can be divided into two types: non-parametric (w.r.t. CAM) and parametric methods. In this section, we focus on representative works from these two types. Please refer to [19] for a more comprehensive survey on the WSOL task.

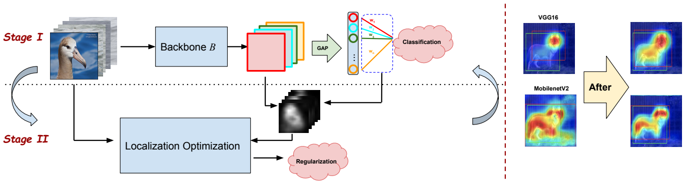

Figure 1: Overview of our MinMaxCAM. Stage I and II optimize the model iteratively via the classification task and our proposed regularizations respectively. After regularizing the model, the localization map can overcome either the under-estimation problem (right top) or the over-estimation problem (right bottom).

[1, 3, 10, 15, 18, 22] and our work belong to the first type. These methods do not need extra networks during the training and inference phases. This makes the methods lightweight, easy to implement and saves computation resources. [15] forces the neural network to focus on other relevant regions of the objects of interest by randomly occluding some patches of the input image when the network is trained for a classification task. Recently, [1] proposes to occlude one image with two complementary occlusion patterns, creating input image pairs, to tackle the WSOL task. [18] extends this idea by using patches from other images as occluding regions in a given image. [3] proposes a simple but effective method: randomly drop out the most highly activated region or apply an attention mask on the feature maps when the classification network is trained. [22] uses information shared by two images from the same class to improve the localization map. They apply two constraints to improve the quality of the localization map. The first constraint is to learn the consistent features of two images of the same class by randomly sampling the features located in the most activated region and minimizing their distance. The second constraint is to compensate the limitation that features can only keep consistency within batches, where it learns a global class center for each class. To increase the coverage of the localization map, [10] proposes to erase the most discriminative regions in the feature map when training a classification task. [17] found that localization maps generated by CAM also activate on the background. To suppress these activations, they propose to compute all the possible CAMs first, and then combine them via a combination function. This combination function is not learned during the training but pre-defined and related to the prediction probability of each possible class. Differently, our method only computes the localization map once for each image.

[7, 20, 21] add extra components based on the CAM model. [20] proposes a two-head architecture where the activation map generated by one stream is used to suppress the most discriminative region of the activation map generated by another one, which is similar to [10]. By doing this, the model learns to use information from other relevant regions instead of the most discriminative one. [21] proposes to generate self-produced guidance (SPG) masks that separate the target object from the background. The masks are learned by the high-confidence regions within the attention maps progressively and they also provide the pixel-level supervisory signal for the classification networks. [7] adds a regressor which takes pseudo-locations generated by the model to learn the coordinates of the object. In addition, two novel losses are proposed to keep the localization map cover the whole object during the training process. Instead of using CAM-based methods, [9] uses a encoder-decoder architecture to learn the location of objects, by leveraging the geometry constrains of objects. A novel loss function that considers the object's geometrical shape is proposed.

Different from these methods, we focus on the linear combination part of the CAM method rather than the feature extraction part, or the structure of the input images. The

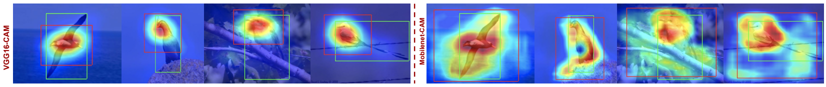

Figure 2: CAM localization maps generated with the VGG16 (left) and MobileNet (right) backbones with their estimated bounding boxes (red) and ground-truth (green).

linear layer is optimized based on the proposed regularizations, which provides the optimal combination factors to generate the localization map. Similar to the non-parametric methods, our method does not have more trainable parameters. We only introduce two more hyper-parameters which are the weights for the two regularizations.

3 Methodology

3.1 Problem statement

Class Activation Map (CAM) [23] is widely used to localize an object of interest in an image, in a weakly supervised manner. Given a backbone B applied to input image I, followed by GAP and a linear layer fc for prediction, in which $w \in \mathbb{R}^{C \times K}$ is the weight matrix, where C is the number of classes, the CAM of an image is computed as:

$$CAM_{raw} = \sum_{k=0}^{K-1} w_k^c B(I) \tag{1}$$

where c is the class of image I. In short, the localization map (CAM) is a linear combination of the feature maps of the last convolutional block of the backbone B. The weights of this combination are taken directly from the weights of the linear layer w.r.t. the class predicted for image I. [5, 23] noted that localization maps generated by VGG often focus on the most discriminative region of the image, rather than on the whole object, i.e. under-estimating the location. In parallel with [17], we observed that for different backbones B, the localization map can sometimes cover the whole image, i.e. over-estimating the location. Fig. 2 shows some examples of the two cases. Current methods solve either one of the two problems. Here we propose two regularizations that can address these problems within a single model.

3.2 Common region regularization

The idea of the *Common region regularization (CRR)* is that different images depicting the foreground objects of the same class should share very similar features from the common object. To obtain this object-specific representation from the whole image I, we first acquire the localization map H via Eq. 1. We freeze the trainable parameters of B (noted as B^* for clarity purpose) so that only fc can be updated by H. Then we extract the object-specific feature f via $g(B^*(I \odot H))$, where g and $\odot$ refer to GAP and element-wise multiplication respectively. Given S different images from the same class, we have

$$CRR = \frac{1}{S(S-1)} \sum_{i=0}^{S-1} \sum_{j=0}^{S-1} ||f_i - f_j||_2^2 \tag{2}$$

CRR calculates the pair-wise distance of the features of $I \odot H$. By minimizing CRR, we **minimize** or remove activations on the localization map by suppressing activations in non-object regions of the images by updating fc.

There are two conditions in place for CRR to work: i) the set of images from the same class should have different background, and ii) f should have different activation values for different backgrounds. The first assumption is dependent on the dataset. We will discuss the second assumption later in Sec. 4.3.

3.3 Full region regularization

For the case where the localization map H under-estimates the object region, i.e. H focuses on the most discriminative region, we propose *Full region regularization (FRR)* to enlarge the localization map. In addition, *FRR* can also compensate a possible side-effect of using *CRR*, that is B only focusing on a very small part of the object if the objects from the same class have some inner difference in different images.

$$FRR = \frac{1}{S}\sum_{i=0}^{S-1} ||f_i - f_i^o||_2^2 \tag{3}$$

f_i is the object-specific feature, which is defined in Sec. 3.2. f^o is the feature of the original image I, i.e. $f^o{=}g(B^*(I))$. Notably, f^o cannot update the model. *FRR* calculates the distance between the object-specific feature f and the image feature f^o. Minimizing *FRR* makes f_i closer to f^o, hence H will change towards the identity matrix $\mathbf{1}$ since $B^*(I)$ can be also interpreted as $B^*(I \odot \mathbf{1})$. In other words, Minimizing *FRR* has the effect of **maximizing** the activations on the localization map. Similar to *CRR*, there is one assumption in place for *FRR* to work: B should not be invariant to changes in intensity. We will discuss it in Sec.4.3.

3.4 Training process

The training process has two stages. For Stage I, it takes $N \times S$ images as input, where N is the number of the set. The model is trained for the classification task, i.e. update the backbone B and linear layer fc via the cross-entropy loss. It is the same as CAM. Stage I is important because it guarantees the localization map H to be object-centered.

$$L_{S1} = -\sum_{i=1}^{N\times S} c_i log(\hat{y}_i) \tag{4}$$

For stage II, the backbone is frozen and used as a feature extractor, noted as B^*. It receives the images multiplied by the localization map ($I \odot H$) as input to get the object-specific features. H is obtained via Eq. 1 and can only drive the update of fc since the rest of the architecture, i.e. the backbone, is frozen. This step introduces an intensity change on the original image. We discuss its effects in Sec. 4.3. The features (f and f^o) extracted by B^* are used for the two proposed regularizations. By minimizing *CRR* and *FRR*, the loss updates fc, making H gradually more accurate by adjusting the combination weights (w_k^c).

$$L_{S2} = \lambda_1 CRR + \lambda_2 FRR \tag{5}$$

L_{S1} and L_{S2} update the model every mini-batch. Fig. 3 shows the proposed method. To train our model there is no extra hyper-parameters besides the weights for the two regularizations. During testing, the localization map is generated via CAM (Eq. 1), therefore, there is no need for a set of images per class.

Why freeze the backbone (B) in Stage II? The goal of stage II is to optimize the weights w_k^c which are used to generate the localization map by minimizing CRR and FRR. The backbone serves as a feature extractor in this stage. If the backbone was also updated, after minimizing *FRR*, it would become invariant to intensity changes. In the later training process, it cannot not measure the difference between f^o and f.

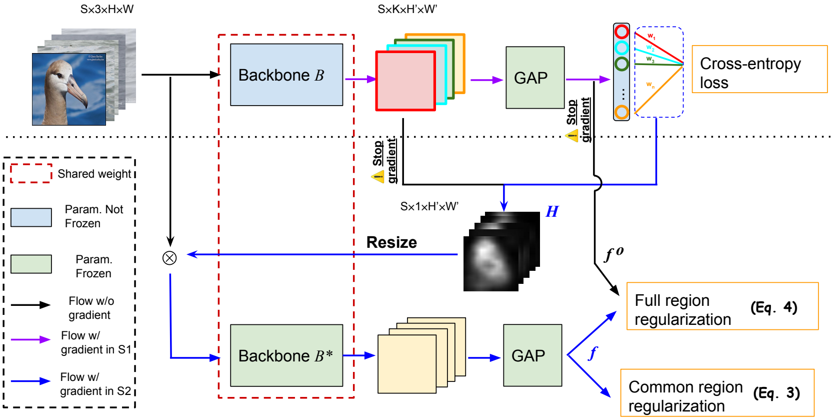

Figure 3: The diagram of our proposed method. In Stage I (above the dashed line) we train B and fc for a classification task. In Stage II (below) we multiply the localization map H with the images to extract the object-specific features and compute *CRR* and *FRR*, where B is frozen (noted as B^*). The two regularizations update the weights of fc (the blue arrow).

4 Experiments

4.1 Evaluation Protocol

Datasets We consider three widely used datasets ImageNet [6], CUB-200-2011 (CUB) [16] and OpenImages instance segmentation subset [2, 5] to evaluate our method. ImageNet contains 1,000 classes with over 1 million images. Following [5], we use ImageNetV2 [12] as the validation set to tune our model. This validation set contains 10 images per class with the object bounding boxes annotated by [5]. CUB has 200 fine-grained classes of birds with 5,994 images for training and 5,794 images for testing. Similarly, we follow [5] to use a validation set collected by them to tune the model. The validation set contains 1,000 images in total, around 5 images per class. OpenImages instance segmentation subset (OpenImages) [4] covers 100 classes. It contains 29,819, 2,500 and 5,000 images for training, validation and testing, respectively. Every image has the object segmentation as annotation.

Performance metric [3, 15, 18, 20, 22, 23] use a pre-defined threshold (0.2) for the generated CAM to produce a localization region. [5] argues that using a fixed pre-defined threshold can be disadvantageous for certain methods since the ideal threshold may depend on the data and architecture that are used. In short, for different datasets, architectures and methods, the ideal threshold is different. We follow this idea and use the metric proposed by [5]. For the ImageNet and CUB datasets we use two threshold-free metrics to evaluate the localization map, i.e. MaxBoxAcc and MaxBoxAccV2. MaxBoxAcc is equivalent to *GT-known localization accuracy* where one localization map is counted correct when the intersection over union (IoU) of the estimation and ground truth bounding box is larger than 0.5. Differently, to avoid using a fixed pre-defined threshold for binarizing the localization map, here we set various τ thresholds to find the best performance. MaxBoxAccV2 is the average of three MaxBoxAcc when the IoU is 0.3, 0.5 and 0.7. For OpenImages, since we have access to the segmentation mask, we use the *pixel averaged precision (PxAP)* proposed by [5]. Similarly, *PxAP* is also threshold-free. Please refer to [5] for more details.

Implementation Details We consider three backbones: VGG16 [14], ResNet50 [8] and the lightweight MobilenetV2 [13]. Following [5, 22], for ResNet50 and MobilenetV2 we

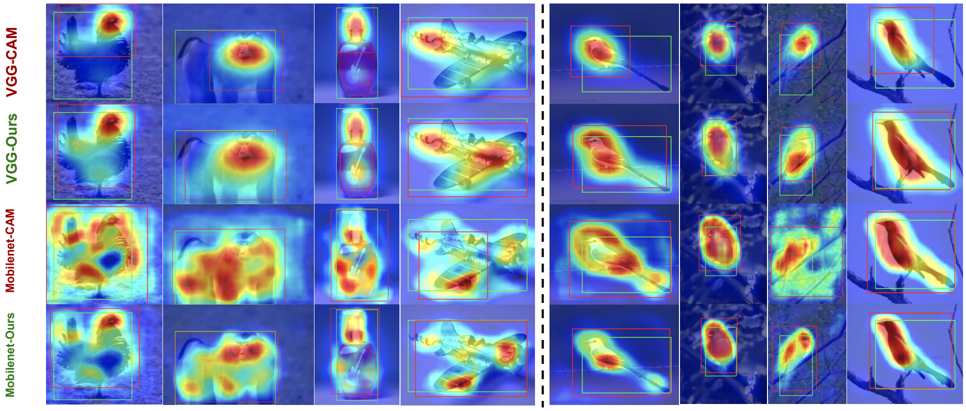

Figure 4: Qualitative comparison of the localization map H on the ImageNet and dataset. For reference we show the ground truth bounding box (green) and the one estimated by H (red) based on the optimal threshold τ.

increase the size of the last feature map by changing the stride of convolution layers from 2 to 1. We set set size S=5, N=12 (i.e. batch size = 60) for all the experiments except those with the ResNet50 backbone, where N=10 due to GPU memory limitations. For τ we set 100 intervals between 0 and 1.

4.2 Comparison with State-of-the-art Methods

In this part, we compare the proposed method w.r.t. several state-of-the-art methods on ImageNet, CUB and OpenImages datasets. Table 1 shows quantitative results. The quantitative results from these methods except I2C are taken from [4, 5] where the authors used the validation set to select the final models. We implement the I2C method and use their suggested hyper-parameters to train the models. The results clearly show that our method outperforms the competing methods on all three datasets except when ResNet50 is selected as backbone for the CUB dataset. In this case, our method is only lower by 0.3 pp, in *MaxBoxAcc* w.r.t. WTL. Interestingly, for ImageNet with a ResNet50 backbone, only our method outperforms CAM. We believe it is due to the proposed *CRR* which minimizes the activations in the background. It is expected that I2C works better when MobilenetV2 or ResNet50 are used as backbone, since the constraints proposed by I2C prevent the model from activating highly in background regions, which is a weakness that Mobilenet and ResNet suffer from. Please note the performance can be further improved when the optimal hyperparameters are found.

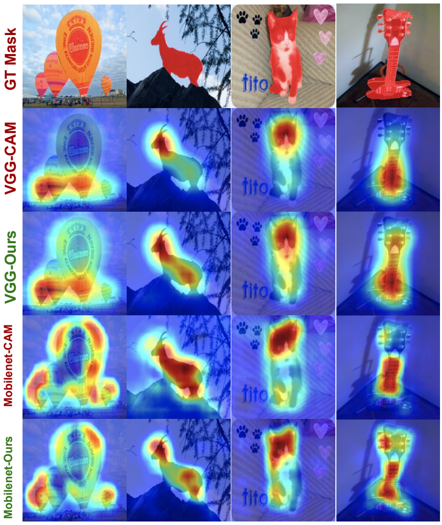

Figure 5: Qualitative comparison of the localization map H on OpenImages dataset. The first row shows the input image with the target segmentation mask.

Fig. 4 and Fig. 5 show qualitative comparisons w.r.t. CAM on the ImageNet, CUB and

Method	Backbone	ImageNet		CUB		OpenImages
		MaxBoxAcc (%)	MaxBoxAccV2 (%)	MaxBoxAcc (%)	MaxBoxAccV2 (%)	PxAP (%)
CAM [23]	VGG16	61.1	60.0	71.1	63.7	58.1
HaS [15]	VGG16	0.7	0.6	5.2	0	-1.2
ACoL [20]	VGG16	-0.8	-2.6	1.2	-6.3	-3.4
SPG [21]	VGG16	0.5	-0.1	-7.4	-7.4	-2.2
ADL [3]	VGG16	-0.3	-0.2	4.6	2.6	0.2
CutMix [18]	VGG16	1.0	-0.6	0.8	-1.4	0.1
I2C [22]	VGG16	-	-	-2.7	-3	-1
WTL [1]	VGG16	2.3	-	6.4	-	–
Ours	VGG16	***3.5***	***2.4***	***12.8***	***6.5***	***1.9***
CAM [23]	ResNet50	64.2	63.7	73.2	63.0	58.0
HaS [15]	ResNet50	-1	-0.3	4.9	1.7	0.2
ACoL [20]	ResNet50	-2.5	-1.4	-0.5	3.5	-0.2
SPG [21]	ResNet50	-0.7	-0.4	-1.8	-2.6	-0.3
ADL [3]	ResNet50	0	0	0.3	-4.6	-3.7
CutMix [18]	ResNet50	-0.3	-0.4	-5.4	-0.2	-0.7
I2C [22]	ResNet50	-	-	0.3	1.0	**2.9**
WTL [1]	ResNet50	0.6	-	**5.3**	-	–
Ours	ResNet50	***3.9***	***2.5***	*5.0*	***4.8***	***2.9***
CAM [23]	MobilenetV2	60.8	59.5	65.3	58.1	54.9
I2C [22]	MobilenetV2	-	-	1.9	1.5	3.3
Ours	MobilenetV2	***4.5***	***3.8***	***10.5***	***6.9***	***4.4***

Table 1: Quantitative comparison w.r.t. state-of-the-art. The numbers indicate the difference w.r.t. the baseline method CAM. The scores of [3, 15, 18, 20, 21, 23] are taken from [4, 5] while [1] is taken from itself. [22] was computed by ourselves. Due to limited computation resources we limit ourselves to report performance only on CUB and OpenImages.

OpenImages datasets. It clearly shows that our method can enlarge H when it originally focuses on a small region and reduces it when it is highly-activated in the background. In the fourth-column example from the ImageNet dataset generated by the VGG backbone, the effect of different optimal thresholds τ can be noticed. The same effect can be seen in the Mobilenet-based localization map of the last example in the ImageNet dataset. In addition, in some cases although the estimated bounding box of CAM has a large IoU with the ground truth, the object region has a stronger activation for our method (e.g. the last example of the ImageNet).

4.3 Analysis of our method

We analyze several aspects of our method by conducting experiments on the CUB dataset.

Background activation We have introduced in Sec. 3.2 that the backbone should activate differently for different background regions for *CRR* to work. If the assumption holds, then $f \in \mathbb{R}^K$ ($f = g(B^*(I \odot H))$) from the same class should be distributed together and more centered in the K-dim feature space, compared with $f^o \in \mathbb{R}^K$ ($f^o = g(B^*(I))$). To quantitatively measure the statistical dispersion of these representations, for each class, we first $L2$-normalize the representations and calculate their distance to the class center which is calculated by averaging all the representations of that class, then compute the average distance. More centered distribution means a shorter distance between each representation and the class center. Then we compute the average distance for all classes. For f and f^o, the average distance are 1.03 ± 0.34 and 1.47 ± 0.38, respectively, which suggests that f is distributed more centered compared with f^o. The result verifies our assumption that the backbone is activated differently for different backgrounds of the input image.

Intensity change For *FRR* to work, the backbone should *not* be robust to changes in intensity of its input. Otherwise, there would be no difference between f and f^o (Sec. 3.3). The experiment based on the VGG backbone has implied that this assumption holds. Here we conduct another experiment to verify this condition from another direction. We add intensity changes as one of the data augmentations in stage I. In stage II we do not change anything. By doing this, we force B to become *more* robust to intensity changes on its input (Please note, we cannot make B completely robust). We use VGG16 as backbone. The performance drops 2.5pp and 1.9pp for MaxBoxAcc and MaxBoxAccV2, respectively. This shows that indeed the performance drops when B becomes more robust to intensity changes.

Masking inputs vs. masking features In order to get the object-specific representation f, we suggest applying element-wise product between H and the input image I firstly and then send it to the same backbone with the trainable parameters frozen (B^*). It can be argued that H can be applied on the extracted feature map (noted as $f' \in \mathbb{R}^{N\times C\times H\times W}$) from stage I directly, which can avoid the re-computation of the feature. However, this may produce some problems. On the one hand, f' not only has spatial information but also has C channels (around 1024 or 2048 for our backbones), and different channels can represent different concepts of the image [11]. On the other hand, the localization map $H \in \mathbb{R}^{H\times W}$ only contains spatial information, no channel-wise information. Information can be lost if H is directly applied on f' since the activations on the same spatial location but different channels will be encouraged/suppressed in the same scale. To verify our analysis, we conduct an experiment where we apply H directly on f'. Table 3 suggests that our analysis is correct. Especially, for VGG16 the performance is even worse than CAM.

Classification vs. localization The classification task sometimes rely on information from the background [11], while the localization task only focuses on the foreground object. Therefore, a good localization model is not necessarily a good classification model. The evaluation metric *top-1/5 Loc.* takes into account both localization and classification accuracy of a given localization model, therefore, it is not able to accurately measure the localization performance [5]. *Top-1/5 Loc.* can be low because of the classification accuracy even if the localization accuracy is good. In order to compute *Top-1/5 Loc.* fairly, we propose a simple path: train a separate classification model to provide the predicted class for the object localization model. In practice, we use ResNet50 to train the classifier, whose classification accuracy on CUB dataset is 77.3%. Table 2 shows the *Top-1/5 Loc.* results. In order to compare with the competitive methods, here we use MobilenetV1 as backbone. The numbers of the competitive methods are taken from [1, 3, 17].

Method	*Top 1 Loc. / Top 5 Loc.*
VGG-ACoL	45.9 / -
VGG-ADL	52.3 / -
VGG-CCAM	50.1 / 63.8
VGG-WTL	58.12 / -
VGG-Ours	***66.0 / 83.9***
MobilenetV1-HaS	44.7 / -
MobilenetV1-ADL	47.7 / -
MobilenetV1-Ours	***54.8 / 69.4***

Table 2: Top-1 and Top-5 localization rate.

4.4 Ablation study

Effects of *CRR* and *FRR* for different backbones (B) We analyze the effect of *CRR* and *FRR* on different backbones. In practice, intuitively, if the localization map H always localizes the most discriminative region of the object, *FRR* should play a more important role in the training process. On the contrary, *CRR* should be more essential if H is relatively highly activated in background regions. To verify the influence of *CRR* and *FRR*, we gradually increase/decrease the weight of one regularization with the other one fixed.

Fig. 6 shows the performance curve. For VGG16, performance decreases gradually as the weight for *CRR* increases. The opposite occurs with *FRR*. The performance increases and reach the peak when the weight of *FRR* is 10, afterwards the performance decreases slightly

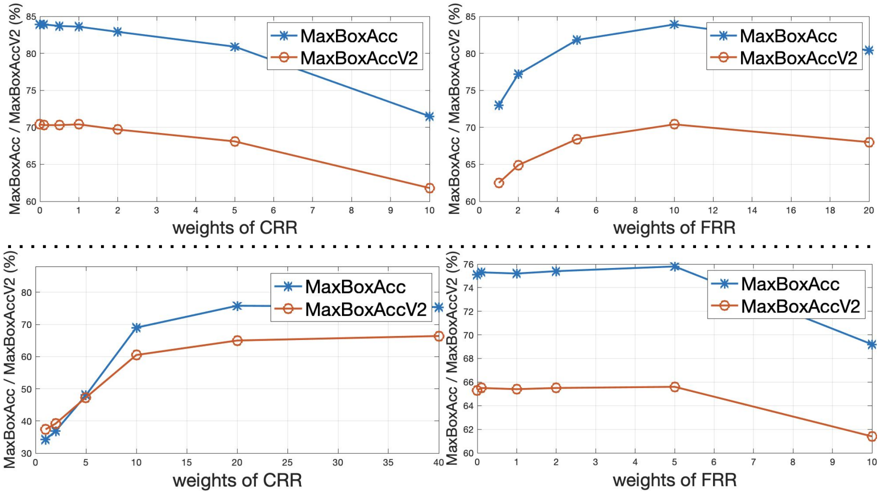

Figure 6: Ablation study w.r.t. *CRR* and *FRR* for the VGG16 (top) and MobilenetV2 (bottom) backbones, respectively. For each plot, we fix one regularization and ablate the other.

(weight=20). On the contrary, increasing the weight of *CRR* boosts the performance for MobilenetV2. The performance decreases slightly when the weight of *FRR* is 0 while it drops dramatically when a larger weight is applied. These trends are expected. VGG16 focuses on small discriminative regions rather than the whole object, hence *FRR* which maximizes the activations on the localization map improves the object localization ability while *CRR* which suppresses the activations makes the model worse. An opposite trend can be observed from MobilenetV2 which activates frequently on the background. The ablation study confirms the effectiveness of our proposed *FRR* and *CRR*.

Effect of the set size (S) Here we discuss how the set size S influences the performance of *CRR*. *CRR* can benefit from using a relatively large S because more representations can be grouped together simultaneously. We conduct series of experiments, where we decrease S from 5 to 2 gradually. We use Mobilenet as backbone and keep the batch size the same (60 images). The results in Table 4 indicate that for a larger set size S, *CRR* indeed works better.

Method	*MaxBoxAcc* (%)	*MaxBoxAccV2* (%)
VGG-Ours	83.9	70.2
Apply H on f'	-19.1	-12.5
MobilenetV2-Ours	75.8	65.0
Apply H on f'	-4.1	-3.0

Table 3: Masking inputs vs.masking features

Set size S	*MaxBoxAcc* (%)	*MaxBoxAccV2* (%)
S=5	75.8	65.0
S=4	74.7	64.6
S=3	74.4	64.4
S=2	73.9	63.9
CAM	65.3	58.1

Table 4: Effect of the set size S.

5 Conclusion

We propose two representation regularizations, *Common Region Regularization* and *Full Region Regularization*, to overcome the weaknesses of CAM-based weakly supervised object localization methods. Our method relies only on a standard classification model; no extra network is needed. Through extensive analysis, we discuss relevant aspects of our method and show that it is capable of surpassing the state-of-the-art by a significant margin.

Acknowledgement This work is founded by FWO Fundamental Project (G0A4720N) "Design and Interpret: A New Framework for Explainable Artificial Intelligence".

References

[1] Sadbhavana Babar and Sukhendu Das. Where to look?: Mining complementary image regions for weakly supervised object localization. In *Winter Conference on Applications of Computer Vision (WACV)*, 2021.

[2] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. *Computer Vision and Pattern Recognition (CVPR)*, 2019.

[3] Junsuk Choe and Hyunjung Shim. Attention-based dropout layer for weakly supervised object localization. In *Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] Junsuk Choe, Seong Joon Oh, Sanghyuk Chun, Zeynep Akata, and Hyunjung Shim. Evaluation for weakly supervised object localization: Protocol, metrics, and datasets. *arXiv preprint arXiv:2007.04178*, 2020.

[5] Junsuk Choe, Seong Joon Oh, Seungho Lee, Sanghyuk Chun, Zeynep Akata, and Hyunjung Shim. Evaluating weakly supervised object localization methods right. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

[7] Guangyu Guo, Junwei Han, Fang Wan, and Dingwen Zhang. Strengthen learning tolerance for weakly supervised object localization. In *Computer Vision and Pattern Recognition (CVPR)*, 2021.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, 2015.

[9] Weizeng Lu, Xi Jia, Weicheng Xie, Linlin Shen, Yicong Zhou, and Jinming Duan. Geometry constrained weakly supervised object localization. In *European Conference on Computer Vision (ECCV)*, 2020.

[10] Jinjie Mai, Meng Yang, and Wenfeng Luo. Erasing integrated learning: A simple yet effective approach for weakly supervised object localization. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[11] José Oramas M, Kaili Wang, and Tinne Tuytelaars. Visual explanation by interpretation: Improving visual feedback capabilities of deep neural networks. In *International Conference on Learning Representations (ICLR)*, 2019.

[12] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? *International Conference on Machine Learning (ICML)*, 2019.

[13] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

[15] Krishna Kumar Singh and Yong Jae Lee. Hide-and-Seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *International Conference on Computer Vision (ICCV)*, 2017.

[16] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[17] Seunghan Yang, Yoonhyung Kim, Youngeun Kim, and Changick Kim. Combinational class activation maps for weakly supervised object localization. *Winter Conference on Applications of Computer Vision (WACV)*, 2020.

[18] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *International Conference on Computer Vision (ICCV)*, 2019.

[19] Dingwen Zhang, Junwei Han, Gong Cheng, and Ming-Hsuan Yang. Weakly supervised object localization and detection: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, abs/2104.07918, 2021.

[20] Xiaolin Zhang, Yunchao Wei, Jiashi Feng, Yi Yang, and Thomas Huang. Adversarial complementary learning for weakly supervised object localization. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[21] Xiaolin Zhang, Yunchao Wei, Guoliang Kang, Yi Yang, and Thomas Huang. Self-produced guidance for weakly-supervised object localization. In *European Conference on Computer Vision (ECCV)*, 2018.

[22] Xiaolin Zhang, Yunchao Wei, and Yi Yang. Inter-image communication for weakly supervised localization. In *European Conference on Computer Vision (ECCV)*, 2020.

[23] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.