CREAM: Weakly Supervised Object Localization via Class RE-Activation Mapping

Jilan Xu¹ Junlin Hou¹ Yuejie Zhang¹ Rui Feng¹ Rui-Wei Zhao² Tao Zhang³ Xuequan Lu⁴ Shang Gao⁴
¹School of Computer Science, Shanghai Key Lab of Intelligent Information Processing, Shanghai Collaborative Innovation Center of Intelligent Visual Computing, Fudan University
²Academy for Engineer and Technology, Fudan University
³Shanghai University of Finance and Economics ⁴Deakin University
{jilanxu18,jlhou18,yjzhang,fengrui,rwzhao}@fudan.edu.cn, taozhang@mail.shufe.edu.cn, {xuequan.lu,shang}@deakin.edu.au

Abstract

Weakly Supervised Object Localization (WSOL) aims to localize objects with image-level supervision. Existing works mainly rely on Class Activation Mapping (CAM) derived from a classification model. However, CAM-based methods usually focus on the most discriminative parts of an object (i.e., incomplete localization problem). In this paper, we empirically prove that this problem is associated with the mixup of the activation values between less discriminative foreground regions and the background. To address it, we propose Class RE-Activation Mapping (CREAM), a novel clustering-based approach to boost the activation values of the integral object regions. To this end, we introduce class-specific foreground and background context embeddings as cluster centroids. A CAM-guided momentum preservation strategy is developed to learn the context embeddings during training. At the inference stage, the re-activation mapping is formulated as a parameter estimation problem under Gaussian Mixture Model, which can be solved by deriving an unsupervised Expectation-Maximization based soft-clustering algorithm. By simply integrating CREAM into various WSOL approaches, our method significantly improves their performance. CREAM achieves the state-of-the-art performance on CUB, ILSVRC and OpenImages benchmark datasets. Code will be available at https://github.com/Jazzcharles/CREAM.

1. Introduction

Weakly Supervised Object Localization (WSOL) aims to localize objects only belonging to one class in each image using image-level supervision [6, 33, 34, 38]. WSOL alleviates massive efforts in obtaining fine annotations.

Prior works mainly follow the pipeline of training a classification network then deriving the Class Activation Mapping (CAM) [38] for WSOL. The foreground region is determined by the CAM values larger than the threshold. However, CAM only highlights the most discriminative regions (i.e., incomplete localization). Existing works seek to discover the complete object via adversarial erasing [22, 34], spatial regularization [15, 19] or attention mechanism [6]. Most of them still acquire the activation maps via a classification pipeline like CAM to answer “which pixels contribute to the class prediction”.

In this paper, we argue that the incomplete localization
problem of CAM is associated with the mixup of the activation values between less discriminative foreground regions and the background. Figure 1 (top) illustrates the distributions of foreground activations (red) and background activations (blue) in CAM. High activations are only dominated by the most discriminative parts (e.g., bird head). In contrast, numerous low activations belong to both the less discriminative parts (e.g., bird body) and the background, making them hard to distinguish. The mixup activations bring the challenge of balancing the precision and the recall of the foreground object. Specifically, the widely adopted threshold $\tau=0.2$ [6, 22, 38] appears too large to completely cover foreground object. Whereas tuning a very small threshold (e.g., $\tau<0.1$) may induce significant false-positive localizations. Moreover, due to mixup activations, a slight change to the threshold leads to a drastic change to the IOU when $\tau$ is small (red curve), and therefore the localization result is sensitive to the threshold. Choe et al. [5] also revealed that a well chosen threshold could lead to a misconception of improvement over CAM. This indicates the infeasibility of improving CAM by simply controlling the threshold.

To address the above challenges, we propose Class RE-Activation Mapping (CREAM), a novel framework for WSOL. We reframe the WSOL task by answering "whether the pixel is more similar to the foreground or the background". In particular, we introduce class-specific foreground (background) context embeddings describing the common foreground (background) features. During training, a momentum preservation strategy is developed to update the context embeddings under the guidance of CAM. With abundant context information, the learned embeddings serve as initial foreground (background) cluster centroids for each class. During inference, we cast the re-activation mapping as a parameter estimation problem formulated under the Gaussian Mixture Model (GMM) framework. We solve it by deriving an Expectation-Maximization (EM) based soft-clustering algorithm. CREAM boosts the activation values of the integral object, eases the foreground-background separation and shows robustness to the threshold, as shown in Figure 1 (bottom). It outperforms prior WSOL works on CUB, ILSVRC and OpenImages datasets.

To sum up, the contributions of this paper are as follows:

- We propose CREAM, a clustering-based method, to solve the mixup of activations between less discriminative foreground regions and the background by boosting the activations of the integral object regions.

- We devise a CAM-guided momentum preservation strategy to learn the class-specific context embeddings and use them as initial cluster centroids for re-activation mapping.

- We regard re-activation mapping as a parameter estimation problem under the GMM framework and solve it by deriving an EM-based soft-clustering algorithm.

- CREAM achieves state-of-the-art localization performance on CUB, ILSVRC and OpenImages benchmark datasets. It can also serve as a plug-and-play method in a variety of existing WSOL approaches.

2. Related Work

2.1. Weakly supervised object localization

Regressor-free WSOL methods. Following Class Activation Mapping (CAM) [38], most approaches obtain the class prediction and the localization map with a classification network only [2, 6, 22, 27, 35, 37]. To tackle the incomplete localization problem of CAM, several methods adopted an iterative erasing strategy on input images [22, 32] or feature maps [27, 35] to force the network’s attention to the remaining parts of an object. Inspired by the attention mechanism, Zhang et al. [36] leveraged pixel-level similarities across different objects to acquire their consistent feature representations. RCAM [1] offered thresholded average pooling, negative weight clamping and percentile thresholding as ways to improving CAM. SEAT [37] sampled top-K activations as foreground seeds and used them to assign the label for each pixel. Xie et al. [28] generated compact activation maps under the guidance of low-level features. Different from these approaches, our method obtains the activation map via an EM-based soft-clustering mechanism, which naturally relieves the incomplete localization problem.

Regressor-based WSOL methods. This category of work [10, 15, 25, 33] disentangles the WSOL task into image classification and class-agnostic object localization. The intuition is that a model’s high localization performance is accompanied by low classification accuracy in early epochs, and vice versa in late epochs. The disentanglement aims to boost both the localization and classification performance. PSOL [33] first proved that training an additional regression network for bounding box prediction yielded great improvement. The bounding box annotations were obtained using an unsupervised co-localization approach [26]. Lu et al. [15] applied a classifier, a generator and a regressor to impose the geometry constraint for compact object discovery. SLT-Net [10] produced robust localization results with semantic and visual stimuli tolerance strengthening mechanisms. With the assistance of a separate regression network, these works achieve superior localization performance over most regressor-free methods. Our CREAM is applicable to both the regressor-free and regressor-based methods.

2.2. EM-based deep learning methods

Expectation-Maximization (EM) algorithm has been frequently adopted in deep learning in recent works [3, 12, 14, 16, 30, 31]. Hinton et al. [12] introduced EM routing
to group capsules for part-whole relationship construction. Yang et al. [31] designed prototype mixture models for few-shot segmentation. They applied EM algorithm to estimate the models’ mean vectors for query images. Biggs et al. [3] used EM algorithm to learn a 3D shape prior for animal reconstruction. Most prior works jointly optimized the parameters in the EM algorithm and the network’s parameters during training. In comparison, we derive an EM based algorithm during inference as an unsupervised soft-clustering mechanism for foreground-background separation.

3. Methodology

3.1. Revisiting class activation mapping for WSOL

Let \( F \in \mathbb{R}^{d \times h \times w} \) be the feature map in the last convolutional layer, and each \( f_k \) corresponds to the feature map at channel \( k \). \( w_k^c \) is the weight for the \( k \)-th channel with regard to class \( c \). The class activation mapping \( M_c \) is defined as:

\[
M_c = \sum_k w_k^c f_k. \tag{1}
\]

Meanwhile, the class prediction can be re-written as:

\[
S_c = \sum_k w_k^c \sum_{i,j} f_k(i,j) = \sum_{i,j} M_c(i,j). \tag{2}
\]

where \( i \) and \( j \) stand for the spatial location. In this way, solving WSOL with CAM can be interpreted as answering “which pixels contribute to the class prediction”.

To produce the final bounding box/mask for WSOL evaluation, \( M_c \) is then normalized to \([0, 1]\) and thresholded by a pre-defined hyperparameter \( \tau \). Only the CAM values greater than \( \tau \) are considered as foreground pixels. However, the activation values of the less discriminative foreground region and the background are indistinguishable, as shown in Figure 1 (top). CAM is thus prone to the incomplete localization problem.

3.2. Class re-activation mapping

For each image, our goal is to discover its foreground region by mapping each pixel into foreground or background cluster. We start by introducing \( V_{fg}^c \) and \( V_{bg}^c \) as the context embeddings. \( V_{fg}^c \in \mathbb{R}^d \) and \( V_{bg}^c \in \mathbb{R}^d \) represent the common foreground and background context features in class \( c \), respectively. They can also be regarded as two cluster centroids for each class. The ways to learning the embeddings and realizing re-activation based on the learned embeddings are described below. Figure 2 shows the framework of our proposed Class RE-Activation Mapping.

Training stage: context embedding learning. We follow exactly the same training procedure and the cross-entropy loss as CAM [38] except that we maintain the context embeddings additionally. The basic idea of context embedding learning is to use the foreground (background) features of current mini-batch images to update the embeddings. First, we perform hard-thresholding on CAM to obtain the one-hot binary masks \( M_{fg}^c,M_{bg}^c \in \{0,1\}^{h \times w} \) as foreground-background indicators, which are calculated as:

\[
M_{fg}^c = \mathbb{1}(M_c \geq \delta), \quad M_{bg}^c = \mathbb{1}(M_c < \delta), \tag{3}
\]

where \( \mathbb{1}(\cdot) \) is an indicator function; and \( \delta \) is set to 0.2 \times \text{max}(M_c) \) as suggested in [38]. The foreground (background) features can be retrieved by the element-wise multiplication of the original feature \( F \) and the mask \( M_{fg}^c(M_{bg}^c) \). For each sample \((x, c)\), we exploit a momentum preservation strategy on the foreground (background) embeddings using the spatial average of the foreground (background)
features. Suppose \( m \in \{fg, bg\} \), the update of the embeddings with regard to class \( c \) is given by:

\[
V_c^m = \lambda V_c^m + (1 - \lambda) \frac{1}{||M_c^m||_0} \sum_{i=1}^{h} \sum_{j=1}^{w} F_{ij}(M_c^m)_{ij},
\]

where \( F_{ij} \) is the \( F \) value at location \((i, j)\); \( \lambda \) is the momentum coefficient; \( || \cdot ||_0 \) counts the number of non-zero elements. With rich context features, the trained \( V^{fg} \) and \( V^{bg} \) act as the initial cluster centroids for re-activation.

**Inference stage: re-activation mapping.** At the inference stage, we formulate re-activation mapping as a parameter estimation problem under Gaussian Mixture Model (GMM) [20] and solve it using an unsupervised Expectation-Maximization (EM) algorithm [7]. EM algorithm is a generalization of Maximum Likelihood Estimation (MLE) [20] and solves it using an unsupervised parameter estimation problem under Gaussian Mixture Model (GMM) [20] and solve it using an unsupervised Expectation-Maximization (EM) algorithm [7]. EM algorithm is a generalization of Maximum Likelihood Estimation for probabilistic models with latent variables [4].

**Problem Formulation.** For each sample \( x \), the log-likelihood we aim to maximize is given by:

\[
\log p(x|\theta) = \sum_{i=1}^{h} \sum_{j=1}^{w} \log p(x_{ij}|\theta),
\]

where \( \theta = \{a_{fg}, a_{bg}, V_{fg}, V_{bg}\} \) is the model parameter. In particular, we define the model for each pixel \( x_{ij} \) as a probability mixture model of two distributions, i.e., foreground distribution and background distribution:

\[
p(x_{ij}|\theta) = \sum_{m \in \{fg, bg\}} a_{m} p_{m}(x_{ij}|V_{c}^{m}),
\]

where the mixing weights \( a_{fg}, a_{bg} \in [0, 1] \) and \( a_{fg} + a_{bg} = 1 \). The foreground (background) base model \( p_{fg}(\cdot|b_{fg}) \) is designed to measure the similarity between the image features and the learned embeddings. Instead of using the RBF kernel in GMM, the choice of base models is Gaussian function based on cosine similarity for implementation efficiency:

\[
p_{m}(x_{ij}|V_{c}^{m}) = e^{(V_c^{m})^T F_{ij} / \sigma}, \quad m \in \{fg, bg\},
\]

where \( \sigma \) is a scale parameter. Next, we describe the application of EM in solving the mixture model. We define \( Z^{fg}, Z^{bg} \in [0, 1]^{h \times w} \) as the latent variables. \( Z_{ij}^{m} \) represents the probability of belonging to foreground at location \((i, j)\).

**E-step.** In the E-step, current parameters are utilized to evaluate the posterior distribution of the latent variables, i.e., \( p(Z_{ij}^{fg}|x, a_{fg}, V_{fg}) \) and \( p(Z_{ij}^{bg}|x, a_{bg}, V_{bg}) \). In each iteration \( t (1 \leq t \leq T) \), assuming the model parameters are fixed, the latent variables are computed as:

\[
Z_{ij}^{m(t)} = \frac{a_{m} p_{m}(x_{ij}|V_{c}^{m(t)})}{\sum_{m' \in \{fg, bg\}} a_{m'} p_{m'}(x_{ij}|V_{c}^{m'(t)})}, \quad m \in \{fg, bg\}.
\]

From the perspective of soft clustering, Eq. (8) calculates the similarity between each pixel feature and context embeddings (i.e., centroids), and assigns a soft label \((f_{g}/b_{g})\) to each pixel. Different from the random initialization in EM, in our case, the initial embeddings have gained sufficient class-specific features from context embedding learning.

**M-step.** In the M-step, the purpose is to adjust the context embeddings by maximizing the expected log-likelihood of the image features using the computed latent variables. This enables class-specific context embeddings to be image-specific. The new model parameters can be obtained by:

\[
V_{c}^{m(t+1)} = \frac{\sum_{i=1}^{h} \sum_{j=1}^{w} Z_{ij}^{m(t)} F_{ij}}{\sum_{i=1}^{h} \sum_{j=1}^{w} Z_{ij}^{m(t)}}, \quad a_{m}^{(t+1)} = \frac{\sum_{i=1}^{h} \sum_{j=1}^{w} Z_{ij}^{m(t)}}{h \times w},
\]

where \( m \in \{fg, bg\} \). \( V_{fg} \) and \( a_{fg} \) are updated by the weighted mean of the features and the effective number of pixels assigned to the foreground, respectively.

Eqs. (8) and (9) execute alternately until convergence. \( Z^{fg(T)} \) and \( Z^{bg(T)} \) successfully mark the entire object regions. CREAM naturally re-activates the values of the integral object regions by assigning a higher probability of being foreground to them. The consequent benefit of gaining the activation map via feature clustering is that it avoids Global Average Pooling layer’s bias towards small areas [1].

**3.3. Map calibration.**

So far, \( Z^{fg(T)} \) and \( Z^{bg(T)} \) have been serving as object localization results. According to the underlying assumption of CAM, foreground objects correspond to high values in the activation map. However, we cannot tell from Eqs. (7) and (8) whether higher or lower values correspond to the foreground regions in \( Z^{fg(T)} \). To be specific, \( Z^{fg(T)} \) is expected to be the final localization map only if its foreground regions have higher values. Otherwise, \( Z^{bg(T)} \) should be chosen. To deal with this uncertainty, we perform map calibration by leveraging their averaged probabilities of being the foreground, i.e., \( \overline{Z}^{fg} \) and \( \overline{Z}^{bg} \) as:

\[
\overline{Z}^{m} = \frac{1}{||M_{c}^{m}||} \sum_{i=1}^{h} \sum_{j=1}^{w} Z_{ij}^{m(T)} (M_{c}^{f g})_{ij}, \quad m \in \{fg, bg\},
\]

where the foreground region \( M_{c}^{fg} \) can be similarly retrieved under the guidance of CAM as to the training stage. The calibrated foreground map \( M_{c}^{cal} \) is defined as the one with greater averaged probability:

\[
M_{c}^{cal} = \max (Z^{fg(T)} 1(\overline{Z}^{fg} \geq \overline{Z}^{bg}) (Z^{bg(T)} 1(\overline{Z}^{fg} < \overline{Z}^{bg})).
\]

To further reduce the re-activations over background regions, the final re-activation map \( \hat{M}_{c} \) is formulated as the sum of the calibrated foreground map and CAM:

\[
\hat{M}_{c} = M_{c}^{cal} + M_{c}.
\]
4. Experiments

4.1. Experimental setup

Datasets. We evaluate our method on three popular WSOL benchmark datasets, i.e., Caltech-UCSD-Birds-200-2011 (CUB) [24], ILSVRC [8] and OpenImages [5]. CUB is a fine-grained bird dataset with 200 species. The training set and testing set contains 5,994 and 5,794 images, respectively. ILSVRC contains 1.2 million training images and 50,000 validation images of 1,000 classes. Both CUB and ILSVRC datasets provide bounding boxes for WSOL evaluation. OpenImages is a recently proposed benchmark. It consists of 29,819, 2,500 and 5,000 images of 100 categories for training, validation and testing, respectively.

Evaluation metrics. Following [33, 34], we use three threshold-dependent evaluation metrics: Top-1 localization accuracy (Top-1 Loc), Top-5 localization accuracy (Top-5 Loc) and GT-Known localization accuracy (GT-Known).

Implementation details. We implement our method on three backbones pre-trained on ImageNet [8], i.e., VGG16 [21], InceptionV3 [23] and ResNet50 [11]. Following [5,34,35], the input images are resized to 256×256 and randomly cropped to 224×224. We perform random horizontal flip during training. We train 50/6/10 epochs on CUB, ILSVRC and OpenImages, respectively. The initial learning rate is 0.001 and divided by 10 for every 15/2/3 epochs on three datasets, respectively. Notably, we set $10 \times$ scale of the learning rate to the classifier due to its random initialization. The min-max normalization strategy is adopted in all the experiments. The SGD optimizer is used with the batch size of 32. On average, the total iterations $T=2$ is sufficient to cover foreground objects. By default, $\sigma=8$ and $\lambda=0.8$ are set across all datasets. Specifically, we introduce two variants of CREAM in the experiments:

- **CREAM** stands for the base CREAM model that only uses the classification network (i.e., regressor-free) to obtain both the class prediction and the localization result, as shown in Figure 2. Most ablation studies are conducted over CREAM$_{Base}$.

- **CREAM** includes a regressor to predict the final bounding box following recent regressor-based methods [10, 33], as illustrated in Figure 3. First, we apply the trained CREAM$_{Base}$ model to generate the pseudo bounding boxes for the training set. We then train a regressor from scratch using the pseudo bounding boxes as the ground truth. The MSE loss is calculated over four coordinates of the box. During inference, the class prediction and object location are derived from the classifier and the regressor, respectively. By disentangling the WSOL task into classification and class-agnostic object localization, CREAM$_{Reg}$ solves the contradiction that the localization accuracy gradually decreases while the classification accuracy increases during training [17, 33].

4.2. Comparison with state-of-the-art methods

Results on CUB. Tables 1, 2 and 4 show the quantitative comparisons with other methods. Regarding MaxboxAccV2 in Table 1, CREAM$_{Base}$ achieves a consistent improvement on all the backbones, e.g., up to 10.5% on
Table 2. Quantitative results on CUB.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Backbone</th>
<th>Localization Accuracy</th>
<th>Top-1 Loc.</th>
<th>Top-5 Loc.</th>
<th>GT-Known</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAM [38]</td>
<td>VGG16</td>
<td>44.15</td>
<td>52.16</td>
<td>56.00</td>
<td></td>
</tr>
<tr>
<td>CutMix [32]</td>
<td>VGG16</td>
<td>43.45</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>ACOL [34]</td>
<td>VGG16</td>
<td>45.92</td>
<td>56.51</td>
<td>54.10</td>
<td></td>
</tr>
<tr>
<td>SPG [35]</td>
<td>VGG16</td>
<td>48.93</td>
<td>57.85</td>
<td>58.90</td>
<td></td>
</tr>
<tr>
<td>HaS-32 [22]</td>
<td>VGG16</td>
<td>49.50</td>
<td>-</td>
<td>71.60</td>
<td></td>
</tr>
<tr>
<td>ADL [6]</td>
<td>VGG16</td>
<td>52.36</td>
<td>-</td>
<td>75.40</td>
<td></td>
</tr>
<tr>
<td>DANN [29]</td>
<td>VGG16</td>
<td>52.32</td>
<td>61.96</td>
<td>67.70</td>
<td></td>
</tr>
<tr>
<td>MEIL [17]</td>
<td>VGG16</td>
<td>57.46</td>
<td>-</td>
<td>73.80</td>
<td></td>
</tr>
<tr>
<td>SPA [19]</td>
<td>VGG16</td>
<td>60.27</td>
<td>72.50</td>
<td>77.29</td>
<td></td>
</tr>
<tr>
<td>RCAM [1]</td>
<td>VGG16</td>
<td>61.30</td>
<td>-</td>
<td>80.72</td>
<td></td>
</tr>
<tr>
<td>GC-Net [15]</td>
<td>VGG16</td>
<td>63.24</td>
<td>75.54</td>
<td>81.10</td>
<td></td>
</tr>
<tr>
<td>PSOL [33]</td>
<td>VGG16</td>
<td>66.30</td>
<td>84.05</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SLT-Net [10]</td>
<td>VGG16</td>
<td>67.80</td>
<td>-</td>
<td>87.60</td>
<td></td>
</tr>
<tr>
<td>ORNet [28]</td>
<td>VGG16</td>
<td>67.74</td>
<td>80.77</td>
<td>86.20</td>
<td></td>
</tr>
<tr>
<td>FAM [18]</td>
<td>VGG16</td>
<td>69.26</td>
<td>-</td>
<td>89.26</td>
<td></td>
</tr>
<tr>
<td>CREAMReg</td>
<td>VGG16</td>
<td>70.44</td>
<td>85.67</td>
<td>90.98</td>
<td></td>
</tr>
<tr>
<td>DANN [29]</td>
<td>InceptionV3</td>
<td>49.45</td>
<td>60.46</td>
<td>67.00</td>
<td></td>
</tr>
<tr>
<td>ADL [6]</td>
<td>InceptionV3</td>
<td>53.04</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SPA [19]</td>
<td>InceptionV3</td>
<td>53.59</td>
<td>66.50</td>
<td>72.14</td>
<td></td>
</tr>
<tr>
<td>TC [36]</td>
<td>InceptionV3</td>
<td>55.99</td>
<td>68.34</td>
<td>72.60</td>
<td></td>
</tr>
<tr>
<td>PSOL [33]</td>
<td>InceptionV3</td>
<td>65.51</td>
<td>83.44</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>SLT-Net [10]</td>
<td>InceptionV3</td>
<td>66.10</td>
<td>-</td>
<td>86.50</td>
<td></td>
</tr>
<tr>
<td>FAM [18]</td>
<td>InceptionV3</td>
<td>70.67</td>
<td>-</td>
<td>87.25</td>
<td></td>
</tr>
<tr>
<td>TS-CAM [9]</td>
<td>Deit-S</td>
<td>71.30</td>
<td>83.80</td>
<td>87.70</td>
<td></td>
</tr>
<tr>
<td>CREAMReg</td>
<td>InceptionV3</td>
<td>71.76</td>
<td>86.37</td>
<td>90.43</td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Quantitative results on ILSVRC.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Backbone</th>
<th>CUB</th>
<th>ILSVRC</th>
<th>Localization Accuracy</th>
<th>Top-1 Loc.</th>
<th>Top-5 Loc.</th>
<th>GT</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCAM [1]</td>
<td>ResNet50</td>
<td>59.53</td>
<td>77.58</td>
<td>49.42</td>
<td>62.20</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSOL [33]</td>
<td>ResNet50</td>
<td>70.68</td>
<td>-</td>
<td>53.98</td>
<td>65.44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAM [18]</td>
<td>ResNet50</td>
<td>73.74</td>
<td>85.73</td>
<td>54.46</td>
<td>64.56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREAMReg</td>
<td>ResNet50</td>
<td>76.03</td>
<td>89.88</td>
<td>55.66</td>
<td>69.31</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. Quantitative results using ResNet50.

Results on ILSVRC. Tables 1, 3 and 4 contain the results on ILSVRC. ILSVRC is more challenging than CUB because of its 1,000 classes and multiple salient objects in an image, resulting in potential false-positive localizations. Despite the difficulties, our method respectively exceeds CAM by 9.3% and 6.4% on VGG16 and InceptionV3. Besides, CREAMBase outperforms CAM by an average of 5.1% on MaxBoxAccV2. In terms of Top-1 Loc., our method achieves a modest improvement of 0.3%, 0.3% and 1.2% over the second best methods using VGG16 and InceptionV3 and ResNet50, respectively.

Results on OpenImages. Regarding pixel-wise WSOL evaluation, CREAMBase also achieves high localization performance. As listed in Table 1, when the backbone is ResNet50, CREAMBase boosts by 6.2% and 3.8% over vanilla CAM and the previous best-performed model RCAM [1], respectively. We attribute the gain to the pixel-level EM clustering in CREAM.

4.3 Ablation study

In Eq. (12), we combine the calibrated foreground map \( M_{cal} \) with CAM (\( M_c \)) to get the final re-activation map CREAMBase at the inference stage. In Table 5, we compare CAM (row 1), calibrated foreground map (row 2), CREAMBase (row 3), and CREAMReg (row 4). The calibrated foreground map outperforms the vanilla CAM by a large margin, 29.8% and 5.3% on CUB and ILSVRC datasets, respectively. It demonstrates the effectiveness of the proposed re-activation mapping. By combining
The resolution of the regression network remains a co-supervised method DDT [26] was used to generate pseudo bounding boxes for the training set before a separate regression network. Following PSOL, we also adopt the resolution of 448 × 448 box generation. With the aid of the additional separate regression network, CREAMReg consistently improves CREAMBase on both datasets.

5. Discussion

Pseudo bounding box generation. In PSOL [33], a co-supervised method DDT [26] was used to generate pseudo bounding boxes for the training set before a separate classification-regression model was trained. For comparison, we replace DDT with CREAMBase for bounding box generation. Following PSOL, we also adopt the resolution of 448 × 448 when generating boxes. The input image resolution of the regression network remains 224 × 224. As listed in Table 6, CREAMBase surpasses DDT by a large margin. On ResNet50 backbone, a significant improvement of 15.6% on CUB is observed. Figure 4 shows the examples of the generated pseudo bounding boxes. DDT fails to capture the entire birds when they have long wings or tails, whereas CREAMBase generates accurate boxes regardless of the changes in bird scale. However, a potential drawback is that both CREAMBase and DDT have difficulty in dealing with the reflection, as shown in the last column.

The regression networks in prior works directly predict the coordinates of the bounding box without improving the localization performance. They have difficulties in fitting pixel-wise WSOL evaluation (e.g., on OpenImages). Notably, our proposed CREAMBase improves the activation map before the pseudo box generation in CREAMReg, making CREAM applicable to both bounding box and pixel-wise evaluation.

Iterations \(T\) and momentum coefficient \(\lambda\). Tables 7 and 8 demonstrate the changes of GT-Known with different iterations \(T\) and momentum coefficients \(\lambda\). Both are conducted on CREAMBase. On both CUB and ILSVRC, CREAM yields the best performance when \(T=2\). CREAM \((T=2)\) outperforms CAM by nearly 30% on CUB with only 4.3 ms longer inference time per image. They have similar training time as context embedding learning is not involved in the back propagation. We visualize the re-activation maps in Figure 5. When \(T=1\), the re-activation map roughly captures the foreground object, but there are still false-positive pixels. When \(T=2\), as E-step and M-step execute in turn, the improved foreground (background) cluster centroids have a more precise indication on foreground (background), and the re-activation map is automatically refined. Nevertheless, we find that the re-activation map tends to oversmooth when \(T\) is large. As shown in Table 8, CREAM is not sensitive to the momentum coefficient. A large \(\lambda\) only performs slightly better. When \(\lambda=1.0\), the context embeddings are equivalent to randomly initialized vectors. In this case, the embeddings include much random noise in the EM process, and the performance of CREAM drops dramatically. This suggests that preserving rich context information in context embeddings helps improve the localization.

Visualization of the localization results. As shown in Figure 5, our method successfully re-activates complete foreground regions in comparison to CAM. On CUB, CREAMBase even embodies the capability of precise localization by highlighting the objects with a clear contour. We attribute this contour-aware localization to the EM-based re-activation. In EM process, the soft label for each pixel feature is solely determined by the similarity between the feature and the context embeddings. The pixel belongs to the foreground as long as it is more similar to the foreground context embedding than that of the background.
Figure 5. Visualization results of the raw image, CAM, CREAM$_{\text{Base}}$ ($T=1$) and CREAM$_{\text{Base}}$ ($T=2$) on CUB, ILSVRC, and OpenImages.

Figure 6. BoxAccV2 under different thresholds on CUB. Solid (dashed) lines show the results of baselines (our method).

The effect of re-activation and generalization ability of CREAM. Following [5], we re-implement CAM, HAS [22] and SPG [35] on BoxAccV2 [5] using VGG16. As illustrated in Figure 6 (left), HAS and SPG perform better than CAM only when $\tau$ is small (e.g. $\tau<0.1$). When $\tau$ is large (e.g. $\tau>0.2$), however, there is no apparent improvement by CAM. In comparison, CREAM$_{\text{Base}}$ curve is above CAM curve for most thresholds. Besides, it is flatter around the peak, showing that CREAM$_{\text{Base}}$ is more robust to the choice of threshold.

We integrate CREAM$_{\text{Base}}$ into HAS and SPG, resulting in HAS.CREAM$_{\text{Base}}$ and SPG.CREAM$_{\text{Base}}$, respectively. As shown in Figure 6 (right), both HAS.CREAM$_{\text{Base}}$ and SPG.CREAM$_{\text{Base}}$ significantly promote the localization performance with an improvement of 10% to 60% under different thresholds. The baselines only perform better when $\tau<0.1$ as their foreground activations gather around 0.1. Whereas the foreground activations in our CREAM-based counterparts appear larger. Their peaks are also more flatter than original methods. The results demonstrate the effectiveness and the generalization ability of CREAM.

6. Conclusion

In this paper, we attribute the incomplete localization problem of CAM to the mixup of the activations between less discriminative foreground regions and the background. We propose CREAM to boost the activation values of the full extent of object. A CAM-guided momentum preservation strategy is exploited to learn the class-specific context embeddings. During inference, re-activation is formulated as a parameter estimation problem and solved by a derived EM-based soft-clustering algorithm. Extensive experiments show that CREAM outperforms CAM substantially and achieves state-of-the-art results on CUB, ILSVRC and OpenImages. CREAM can be easily integrated into various WSOL approaches and boost their performance.

Acknowledgement

This work is supported by National Science and Technology Innovation 2030 - Major Project (No. 2021ZD0114001; No. 2021ZD0114000), National Natural Science Foundation of China (No. 61976057; No. 62172101), the Science and Technology Commission of Shanghai Municipality (No. 21511101203), the Science and Technology Major Project of Commission of Science and Technology of Shanghai (No. 2021SHZDZX0103).
References


[33] Chen-Lin Zhang, Yun-Hao Cao, and Jianxin Wu. Rethinking the route towards weakly supervised object localization. In CVPR, pages 13460–13469, 2020. 1, 2, 5, 6, 7

[34] Xiaolin Zhang, Yunchao Wei, Jiashi Feng, Yi Yang, and Thomas S Huang. Adversarial complementary learning for weakly supervised object localization. In CVPR, pages 1325–1334, 2018. 1, 5, 6


