Weakly-Supervised Butterfly Detection Based on Saliency Map

Ting Zhang[a], Muhammad Waqas[b,c], Yu Fang[a], Zhaoying Liu[a,*], Zahid Halim[d], Yujian Li[a,e], Sheng Chen[f,g]

[a] *Faculty of Information Technology, Beijing University of Technology, Beijing, 100124, China*
[b] *Computer Engineering Department, College of Information Technology, University of Bahrain, 32038, Bahrain.*
[c] *School of Engineering, Edith Cowan University, Perth, 6027, WA, Australia.*
[d] *Department of Computer Science and Engineering, GIK Institute of Engineering Sciences and Technology, Topi, 23640, Pakistan*
[e] *School of Artificial Intelligence, Guilin University of Electronic Technology, Guilin, China*
[f] *School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK*
[g] *Faculty of Information Science and Engineering, Ocean University of China, Qingdao 266100 China*

Abstract

Given the actual needs for detecting multiple features of butterflies in natural ecosystems, this paper proposes a model of weakly-supervised butterfly detection based on a saliency map (WBD-SM) to enhance the accuracy of butterfly detection in the ecological environment as well as to overcome the difficulty of fine annotation. Our proposed model first extracts the features of different scales using the VGG16 without the fully connected layers as the backbone network. Next, the saliency maps of butterfly images are extracted using the deep supervision network with shortcut connections (DSS) used for the butterfly target location. The class activation maps of butterfly images are derived via the adversarial complementary learning (ACoL) network for butterfly target recognition. Then, the saliency and class activation maps are post-processed with conditional random fields, thereby obtaining the refined saliency maps of but-

[*] Corresponding author

Email addresses: zhangtingbjut@foxmail.com (Ting Zhang), engr.waqas2079@gmail.com (Muhammad Waqas), 2656933963@qq.com (Yu Fang), zhaoying.liu@bjut.edu.cn (Zhaoying Liu), zahid.halim@giki.edu.pk (Zahid Halim), liyujian@bjut.edu.cn (Yujian Li), sqc@ecs.soton.ac.uk (Sheng Chen)

terfly objects. Finally, the locations of the butterflies are acquired based on the saliency maps. Experimental results on the 20 categories of butterfly dataset collected in this paper indicate that the WBD-SM achieves a higher recognition accuracy than that of the VGG16 under different division ratios. At the same time, when the training set and test set are 8:2, our WBD-SM attains a 95.67% localization accuracy, which is 9.37% and 11.87% higher than the results of the DSS and ACoL, respectively. Compared with three state-of-the-art fully-supervised object detection networks, RefineDet, YOLOv3 and single-shot detection (SSD), the detection performance of our WBD-SM is better than RefineDet, and YOLOv3, and is almost the same as SSD.

Keywords: Butterfly detection, saliency map, class activation map, weakly-supervised object detection.

1. Introduction

The long-term survival and development of human society are critically affected by biodiversity. With the development of human society, however, the biodiversity decline has become increasingly severe, which is now one of the top ten environmental problems worldwide [1]. Reliable species detection is an essential procedure in carrying out relevant biological research and is a prerequisite for studying biological evolutionary and developmental processes [2]. Insects are the most abundant form of animal life. At present, there are over 1.5 million kinds of insects that have been discovered around the world. Butterflies, which are lepidopteran insects with scales on their wings and liquid-sucking proboscis, are among the most diverse insects. There are more than 18,000 butterflies worldwide, of which approximately 1,200 types are found in China [3]. Butterflies play a crucial role in the research of speciation, community ecology, biogeography, climate change, and plant-insect relationships. The challenging problem is that the detection of butterfly species is quite tricky. The shape, color, texture, and pattern of wings vary among butterflies of different types. Manual recognition and classification of butterfly species require professionally trained recognition specialists with prolonged experience. Moreover, the process of manual identification is exceptionally time-consuming and inefficient.

With the development and application of machine learning, favorable conditions have been created for the fast, accurate automatic detection and recognition of butterfly objects. In general, machine learning-based methods first characterize the butterfly specimen images by manually extracting the image features (color, texture, edges and shape) and then implement automatic detection of butterfly images by integrating statistical learning method [4]. In the real world, the primary demand for butterfly detection is ecological butterfly image detection in natural scenes. Due to the complex environmental background of ecological butterfly images and the various postures and self-protective mimicry of butterflies, significant challenges exist in automatic butterfly specimen detection.

In response to the above problems, we propose a weakly-supervised butterfly object detection model based on a saliency map (WBD-SM) along with class activation map. We collect a butterfly dataset with 20 categories of butterfly and use it to demonstrate the effectiveness of our proposed WBD-SM. Our experimental results indicate that the WBD-SM achieves a recognition accuracy of 89.40%, which represents an improvement by 2.60% over the performance achieved by the VGG16 [5]. The WBD-SM also attains a localization accuracy of 95.67%, which is 9.37% and 11.87% higher than those achieved by the deep supervision network with shortcut connections (DSS) [6] and adversarial complementary learning (ACoL) [7], respectively. Furthermore, compared with state-of-the-art fully-supervised object detection networks, including RefineDet [8], YOLOv3 [9], and single-shot detection (SSD) [10], our WBD-SM is superior over RefineDet and YOLOv3, in terms of detection performance, while its detection performance is almost the same as the SSD. To sum up, the main contributions of our work are as follows.

1. We propose a weakly-supervised butterfly detection method based on a saliency map.

2. We explore to modify the saliency map with the class activation map and then generate the bounding box with the finer saliency map.
3. Experimental results show the proposed method outperforms state-of-the-art fully-supervised methods.

The rest of this paper is organized as follows: Section 2 reviews the related work on object detection, while Section 3 details the structure and learning algorithm of our proposed WBD-SM. Section 4 demonstrates the initial experimental results and analysis. Our conclusions are given in Section 5, where future research directions are also suggested.

2. Related Work

Object detection aims to recognize and localize substantial objects of predefined categories from the images accurately and efficiently. Since 2012, due to the excellent performance achieved by deep convolutional neural networks (CNNs) in classification tasks, researchers have increasingly attracted to study the object detection algorithms based on deep learning. Depending on the presence or absence of a candidate box generation stage, the deep learning-based object detection algorithms can be classified into two-phase and one-phase algorithms [11]. The pioneer algorithm of two-phase object detection is the regions-CNN (R-CNN) based on proposal regions, which combines AlexNet with selective search [12, 13]. It utilizes a search algorithm to initially extract about 2,000 proposal regions, each of which is then normalized and inputted into the CNN one by one for feature extraction. Finally, the features are subjected to support vector machine classification and regional regression. R-CNN has brought a qualitative change to the accuracy of object detection. It represents a milestone in applying deep learning to object detection, which also lays the foundation for deep learning-based two-phase object detection. Subsequently, researchers have proposed models like Fast R-CNN [14], Faster R-CNN [15], and Mask R-CNN [16] in succession based on R-CNN.

With two-phase object detection algorithms, the candidate boxes are extracted from the images initially. Then secondary correction is performed based on the proposal regions to yield the detection results. These algorithms achieve high detection accuracy, but their detection speed is quite low. Some researchers have put forward one-phase object detection algorithms to address the inefficiency of two-phase object detection algorithms. Such type of algorithms does not require the branching of proposal regions. For a given input image, the candidate boxes and categories of objects are regressed directly at multiple positions. These algorithms mainly include the you only look once (YOLO) series [17, 18] and SSD series [19].

By discarding the candidate box extraction branches, YOLOv1 [17] directly implements feature extraction, candidate box classification, and regression in the same branchless deep CNN. It simplifies the network structure and slightly improves the detection speed, thus enabling the deep learning-based object detection algorithm to meet the needs of real-time monitoring tasks with the computing power constraint. Later, in response to its insufficient localization accuracy, Redmon and Farhadi proposed YOLOv2 [18], and YOLOv3 [9] successively. The authors utilized the operations batch normalization, high-resolution classifier, direct target box location detection, and multi-scale training to enhance the model detection accuracy.

Based on the regression idea, SSD [10] effectively applies the concept of multi-scale detection to extract multiple feature maps of different scales for detection. Furthermore, it also borrows the anchor mechanism from the faster R-CNN to preset a fixed number of default boxes with different levels and aspect ratios at each location of the extracted feature maps. The network performs dense sampling directly on the feature maps to obtain candidate boxes for prediction. The authors of [19] adopt a feature fusion technique for the extracted features of different scales. Since the features in each scale have information related to other scales, the fusion adds the connections between feature maps in various layers.

The two-phase and one-phase class object detection approaches have their distinct advantages. The existing models combine these two classes of algorithms to get better performance. For example, RefineDet [8] combines the advantages of the two-phase model with the one-phase model. It consists of two inter-connected modules, i.e., the anchor refinement module and the object detection module. Specifically, the first module filters out the negative anchors to reduce the search space of the classifier and coarsely adjusts the positions of anchors to provide better initialization for the second module. The second module then refines the anchors generated by the first module to improve the prediction accuracy for multi-class labels further.

As aforementioned, the two categories of deep learning-based object detection algorithms have achieved particular successes in dealing separately with detection accuracy and efficiency. Nevertheless, both types of algorithms require manual labeling of the object locations, and they all belong to the fully supervised object detection. With the development of deep learning, demanding requirements have been placed on the quantity and quality of labeled data. Manual labeling increasingly becomes unable to meet this demand, as manual labeling suffers from the unavoidable drawbacks of subjectivity and high cost.

To address this problem, weakly-supervised object detection based on image-level annotation has become a hot research topic. The methods of weakly-supervised object detection can be divided into three classes, i.e., the segmentation-based methods [20], the multiple instance learning (MIL)-based methods [21], and the convolutional feature-based methods [22]. Among them, the convolutional feature-based weakly-supervised object detection is regarded as the mainstream method. Zhou *et al.* [23] replaced the fully connected layer of CNN with global average pooling (GAP), where the localization capability of the convolution unit was retained through class activation mapping, thereby generating class activation maps (CAMs). Subsequently, in response to the mere emphasis on local regions with the standard CAM methods, several researchers adopted a variety of means to obtain more holistic CAMs, which ultimately yielded better results of object detection. However, CAM generally is a detection bottleneck, owning to the limitation of the network classification capability and the lack of boundary recognition ability.

Saliency detection, aiming at highlighting visually salient objects or regions in an image, is widely applied as a pre-processing procedure in various computer vision tasks, such as object detection, image segmentation, and visual tracking. Saliency detection approaches can be roughly divided into two groups, i.e., the bottom-up/top-down network and the side-fusion network [24]. The bottom-up/top-down network first generates hierarchical features layer by layer and then detects the salient objects with the final features, with the examples including SFCN [25], DHSNet and AFNet [26]. The side-fusion network aggregates the multi-layer features of the backbone network, and forms a multi-scale feature for detection, with the representatives of OSVOS [27], NLDF [28], and DSS [6]. Compared with the bottom-up/top-down network, the side-fusion network can achieve higher performance gain in saliency detection.

This paper combines the side-fusion based saliency detection and CAM method to build the WBD-SM. Targeting weakly-supervised detection of butterfly objects, we adopt the saliency detection based on the CAM method to enhance the model's attention to butterfly edge information for attaining more accurate detection. The network only needs to detect the saliency maps (SMs) of butterflies, and fuses the SMs and CAMs. Here, the CAMs have two roles. One is to provide the label information for the saliency map. The other is to distinguish the butterfly from the whole image, helping the saliency map to remove non-butterfly regions. Finally, we can get a more accurate and finer saliency map. Although the WBD-SM accomplishes object localization with two subtasks jointly, no additional annotation is made in either of them. Hence, the proposed model is weakly-supervised. More specifically, we generate the SMs of butterfly images with the trained DSSNet [29], which are used as rough labels containing noise to replace the truth labels of the saliency detection subtask. Thus, in the saliency detection subtask, no annotation of images is performed except for the categorical annotation. At the same time, it is only necessary to provide the class labels in the classification task [30].

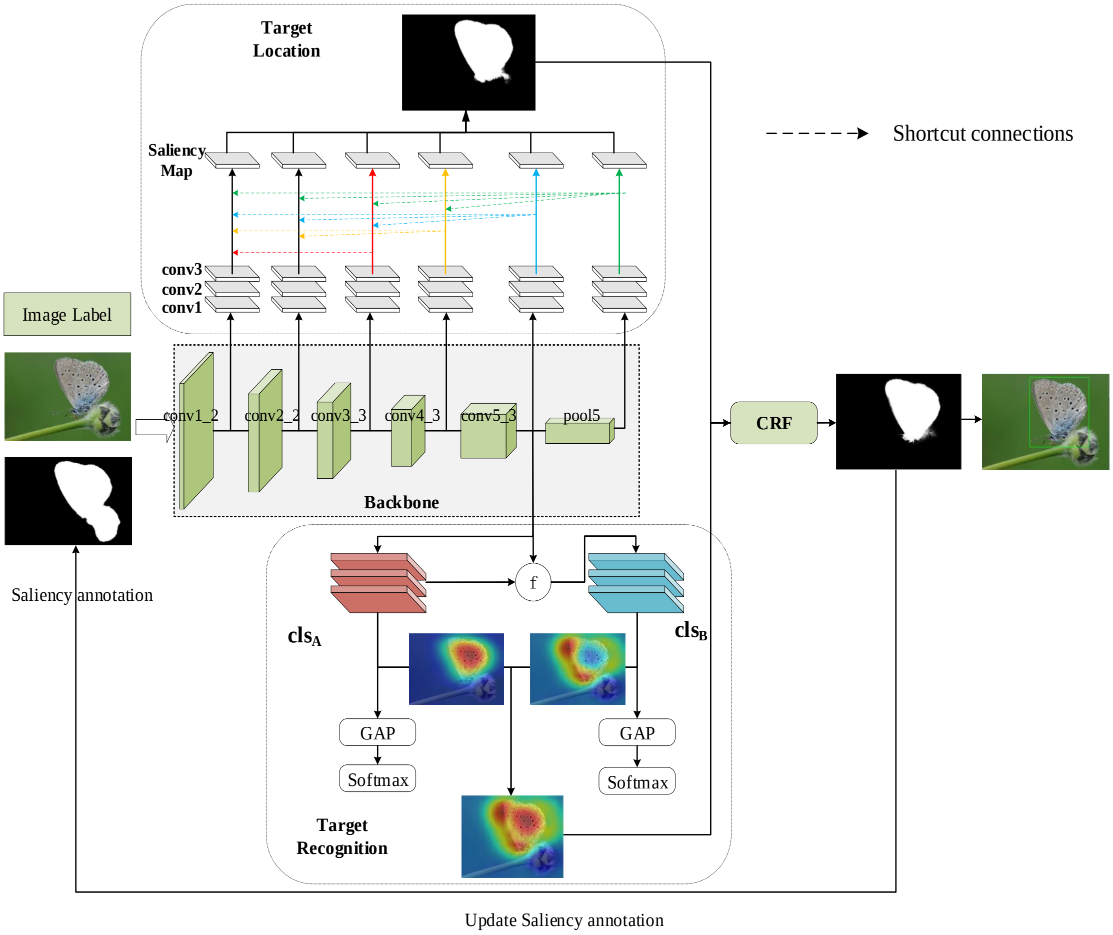


Figure 1: The Structure of WBD-SM.

3. Weakly Supervised Detection Network Based on Saliency Map

3.1. The Overall Architecture

Figure 1 depicts the proposed WBD-SM network, which is a fully convolutional network consisting of the backbone network as well as the target location and recognition networks. The backbone network is the VGG16 [5] without the fully connected layers, while the saliency detection part uses the deep supervision with short connections (DSS) [6] and the recognition part exploits the adversarial complementary learning (ACoL) [7]. Hence, the proposed network offers two types of SMs. One is the general SMs detected by the DSS, and the other is the specific CAMs detected by ACoL.

As shown in Figure 1, the backbone network, i.e., the VGG16 without the fully connected layers, initially extracts the features of the input images. Then, for each layer in the VGG16, the butterflies' locations are identified with the target location network. Furthermore, the types of butterflies are recognized by the target recognition network using the conv5_3 layer of the VGG16 as its input. Finally, the SMs and CAMs are used as the inputs of the conditional random fields (CRFs) to generate the final segmentation maps of butterfly objects, thereby updating the saliency annotation and generating the bounding box.

Specifically, for the backbone network, it is the VGG16 without the fully connected layers, with 13 convolutional layers and 5 pooling layers. It is structured with five blocks of convolutional layers. The first two blocks respectively contain 2 convolutional layers, and the last three blocks include 3 convoliutional layers each. The pooling layer is performed with max-pooling to reduce the size of feature maps, and it has no parameters to learn. Tab. 1 describes the structure of the backbone network in detail.

There are two reasons for selecting VGG16. First, our task is butterfly object detection. It is a task about pixel-level, paying more attention to low-level features. VGG16 has 16 layers and has some low-level features, which are suitable for our task. Second, our dataset has about 2,000 butterfly images, and VGG16 is enough for dealing with this dataset. There is no need to use a more extensive backbone network, such as Inception, Resnet50, and Densenet121. Besides, for this task, we pay more attention to the detection accuracy than the detection time. Therefore, we don't use MobileNet128 [31] as the backbone network either.

There are two reasons to combine the saliency map with the class activation map. One is that saliency detection aims to detect the whole saliency region of the input image, not a specific class of objects. Although it gives nearly accurate boundary information, the saliency map lacks category information, and usually,

Table 1: The Structure of the backbone network.

No.	Layer	input	conv	output
1	conv1_1	224×224×3	64, 3×3	224×224×64
2	conv1_2	224×224×64	64,3×3	224×224×64
3	pooling2	224×224×64	2×2	112×112×64
4	conv2_1	112×112×64	128,3×3	112×112×128
5	conv2_2	112×112×128	128,3×3	112×112×128
6	pooling2	112×112×128	2×2	56×56×128
7	conv3_1	56×56×128	256,3×3	56×56×256
8	conv3_2	56×56×256	256,3×3	56×56×256
9	conv3_3	56×56×256	256,1×1	56×56×256
10	pooling3	56×56×256	2×2	28×28×256
11	conv4_1	28×28×256	512,3×3	28×28×512
12	conv4_2	28×28×512	512,3×3	28×28×512
13	conv4_3	28×28×512	512,1×1	28×28×512
14	pooling4	28×28×512	2×2	14×14×512
15	conv5_1	14×14×512	512,3×3	14×14×512
16	conv5_2	14×14×512	512,3×3	14×14×512
17	conv5_3	14×14×512	512,1×1	14×14×512
18	pooling5	14×14×512	2×2	7×7×512

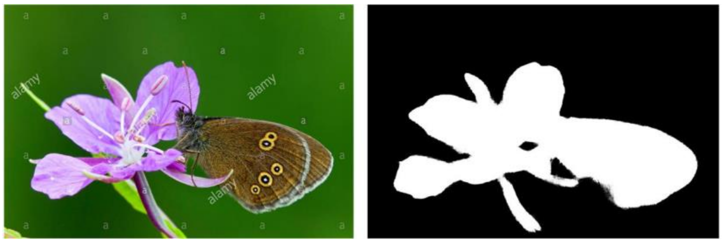


Figure 2: An example of an image and its saliency map.

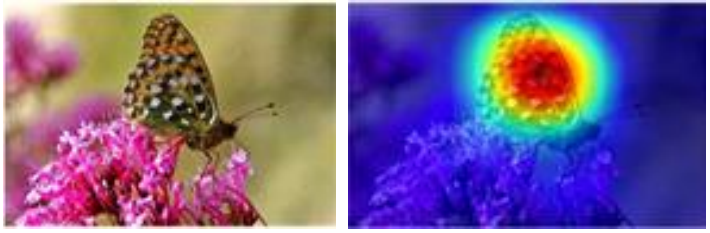


Figure 3: An example of an image and its class activatin map.

the whole saliency region is more significant than that of the target region. Figure 2 displays a butterfly image and its saliency map. The other is the class activation map can locate the general position of the specific target. However, it cannot identify the boundary information. Figure 3 depicts a butterfly image and its class activation map.

3.2. Target Location Network

For a butterfly image, usually, the butterfly is the saliency object. Therefore, we use a saliency detection network, called holistically-nested edge detector (HED) to locate it. As shown in Figure 1, the saliency detection network is

Table 2: The Structure of side output branches.

No.	Layer	conv1	conv2	conv3
1	conv1_2	128,3 × 3	128,3 × 3	1,1 × 1
2	conv2_2	128,3 × 3	128,3 × 3	1,1 × 1
3	conv3_3	256,5 × 5	256,5 × 5	1,1 × 1
4	conv4_3	256,5 × 5	256,5 × 5	1,1 × 1
5	conv5_3	512,5 × 5	512,5 × 5	1,1 × 1
6	pool5	512,7 × 7	512,7 × 7	1,1 × 1

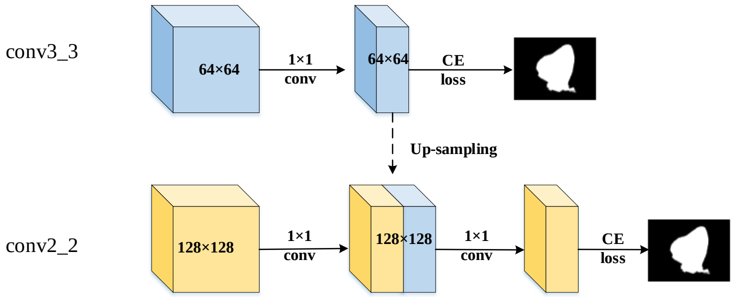


Figure 4: Illustration of shortcut connection from branch conv3_3 to branch conv2_2.

accomplished primarily by the six side-output branches at the upper part of the backbone network. Each branch includes three convolutional layers. Tab. 2 details the structures of these six side-output branches. This part of the network achieves saliency detection by introducing the shortcut connections into the skip structure of HED architecture. The architecture implements short connections and skips connections from the deeper side to the shallower side. Specifically, between the conv3 layer and the saliency map, some horizontal dotted lines across different branches indicate the shortcut connections from the higher branches to the lower branches. These connections utilize the features of the higher branches to guide the lower ones to extract the most salient regions based on the cross-entropy (CE) loss. Figure 4 illustrates the shortcut connection from the conv3 layer of the branch conv3_3 to the conv3 layer of the branch conv2_2.

3.3. Target Recognition Network

For target recognition, it has two tasks. On the one hand, it must classify the butterflies as accurately as possible. On the other hand, it needs to provide supplementary information for the saliency map generated by the target location network to get a finer saliency map. Adversarial complementary learning network uses two adversarial complementary parallel branches, one is trained to learn the most distinguish region, and the other is forbidden to learn the second determined region. By combining these two regions, we can get a class activation map of the butterfly image.

In our target recognition network, we use two branches of A and B to recognize the class labels of a butterfly image to deal with the intra-class variations and between-class similarities. There are two reasons. First, different butterfly images have similar class activation maps, and the most distinguished region usually is not covering the whole regions of the butterfly. Second, the second distinguish region can provide supplementary information for the first distinguish region. As shown in Figure 5, the recognition of butterfly types is accomplished mainly by two adversarial complementary parallel branches, A and B, located at the lower part of the backbone network. Each branch consists of two 3×3 convolutions, a 1×1 convolution, a GAP layer, and the softmax layer. The GAP layer takes the average of each convolutional feature map, and feds to result vector into the softmax layer. There is no parameters to optimize in the global average pooling layer, thus avoiding overfitting.

The network completes the recognition task while generating the CAMs. Among them, branch A utilizes the original feature maps, which can locate the most discriminative region. As for branch B, the feature maps after erasing the most discriminative part (zeroing the corresponding area) are used. Accordingly, the branch is forced to find other features used for classification, which eventually locates the second discriminative region. Through the adversarial learning between branches A and B, the network can identify a more holistic area.

3.4. Objective Function Design

Let the original input image be $\boldsymbol{X}$, the corresponding saliency truth label be $\boldsymbol{Z}$, and the class label be y. The rest of the details are given in subsequent subsections.

3.4.1. Saliency Detection Loss

In the saliency detection, there are a total of six side branches and one fusion layer. For each side branch and fusion layer, the loss function with truth-value needs to be calculated. Suppose that after the m^{th} $(m = 1, \cdots, 6)$ side pass

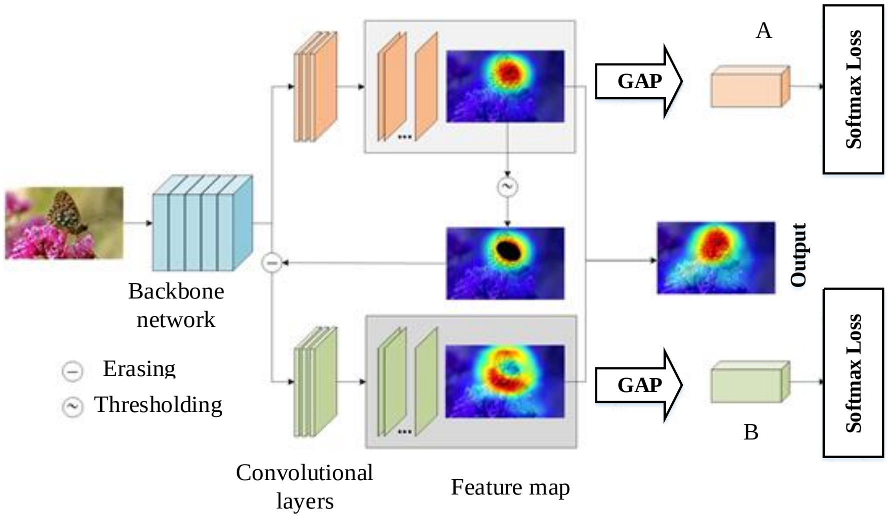


Figure 5: Illustration of the erasing procedure.

through the short and skip connections, the activation value of the output layer is $\boldsymbol{R}^m_{side}$. Then, the loss L^m_{side} of this layer is

$$L^m_{side} = h(\boldsymbol{Z}, \boldsymbol{R}^m_{side}) \tag{1}$$

where $h(\cdot,\cdot)$ denotes the CE loss function of dichotomous classification [32], which is calculated as follows. Let the available data of N samples be given by $\{\boldsymbol{Z}_i, \boldsymbol{R}^m_{side,i}\}_{i=1}^N$, where $\boldsymbol{Z}_i$ is the ith sample of the saliency truth label $\boldsymbol{Z}$ and $\boldsymbol{R}^m_{side,i}$ is the ith sample of the output of the mth branch $\boldsymbol{R}^m_{side}$. Then,

$$h\big(\boldsymbol{Z}, \boldsymbol{R}^m_{side}\big) = -\sum_{i=1}^{N} \boldsymbol{Z}_i \log \boldsymbol{R}^m_{side,i} + (1 - \boldsymbol{Z}_i) \log \big(1 - \boldsymbol{R}^m_{side,i}\big) \tag{2}$$

In both 1 and 2, N is the number of training samples. On the other hand, the loss L_{fuse} of the fusion layer is given by

$$L_{fuse} = h\left(\boldsymbol{Z}, \sum_{m=1}^{6} f_m \boldsymbol{R}^m_{side}\right) \tag{3}$$

where f_m are the weights used during the weighted fusion. Ultimately, the total loss L_S of the saliency subtask is

$$L_S = L_{fuse} + \sum_{m=1}^{6} \alpha_m L_{side}^{m} \tag{4}$$

in which α_m are the weights for the side branch losses. We initialized f_m as 0.167, and α_m as 1 before training. During training, the f_m is constant, and α_m is optimized by gradient descent.

3.4.2. Class Recognition Loss

In the recognition subtask, there are two parallel branches. Suppose that the activation values for the output layers of these two branches are y_a and y_b, respectively. Then, the losses of the two branches are

$$L_a = \bar{h}\left(y, y_a\right) \tag{5}$$

and

$$L_b = \bar{h}\left(y, y_b\right) \tag{6}$$

where $\bar{h}(\cdot, \cdot)$ denotes the CE loss function of polytomous classification [33]. Hence,

$$\bar{h}\left(y, y_a\right) = -\sum_{i=1}^{N} \sum_{c=1}^{C} y_{a,i}^{c} \log y_i^c \tag{7}$$

where y_i^c and $y_{a,i}^c$ are the cth category of the ith samples of y and y_a, respectively, while N is the number of training samples and C is the number of categories. The total loss of class recognition is given by

$$L_C = \beta_a L_a + \beta_b L_b \tag{8}$$

where β_a and β_b denote the weights for the two branch losses. We initialized β_a and β_b as 1, and both of them are optimized by gradient descent.

3.4.3. Multi-task Loss

The network includes two tasks, namely, saliency detection and class recognition. Compared to the saliency detection task, the recognition task is less complicated and easier to train. Thus, if the two subtasks are set with the same weight, the entire training process will be biased towards the recognition task. It isn't easy to balance the two tasks by setting the appropriate weights a priori. To achieve a dynamic finding of the appropriate weight ratio for the multi-task loss, we introduce uncertainty into the loss measurements of different tasks [21]. Specifically, the total loss of the network is defined by

$$L_{final} = \frac{1}{\delta_s^2} L_S + \frac{1}{\delta_c^2} L_C + \log \delta_s + \log \delta_c \tag{9}$$

where δ_s and δ_c denote the noise parameters, which are learnable loss weights. Learning of these two noise weights is based on gradient descent [34]. We initialized δ_s and δ_c as 1, and both of them are optimized by gradient descent.

3.5. Training Process

All of the parameters of WBD-SM were tuned with the backpropagation algorithm, i.e., the parameters of the backbone networks, and that of both the target location network and the target recognition network.

Algorithm 1 displays the algorithm flow of the WBD-SM training process. In the Algorithm 1, $\boldsymbol{W}_S$ and $\boldsymbol{W}_C$ denote the weights of saliency detection and recognition, respectively. In addition, f_S represents the output result of the saliency detection network, f_C represents the output result of the recognition network, and the symbol '$\oplus$' denotes the weighted fusion operation. Furthermore, N_{SM} is the number of training iterations, and we set $N_{\mathrm{SM}} = 25$ empirically.

3.6. Conditional Random Field

After obtaining both the saliency map and the class activation map, we can fuse them to get a finer saliency map. Here, we use the conditional random field to modify its edge. The energy function of conditional random field is

$$E(x) = \sum_p \beta_p(x_p) + \sum_{p,q} \beta_{pq}(x_p, x_q) \tag{10}$$

where x stands for the predictive label of a pixel.

Algorithm 1 WBD-SM training algorithm

```
Input: Training image $\boldsymbol{X}$, saliency label $\boldsymbol{M}$, class label $y$
while the SM is updated less than $N_{\mathrm{SM}}$ do
    if the training converges, then
        Obtain the predicted SM $\boldsymbol{M}_S \leftarrow f_S(\boldsymbol{W}_S, \boldsymbol{X})$
        Obtain the CAM $\boldsymbol{M}_C \leftarrow f_C(\boldsymbol{W}_C, \boldsymbol{X})$
        Update the saliency label $\boldsymbol{M}_{update} = CRF(\boldsymbol{M}_S \oplus \boldsymbol{M}_C)$
    end if
end while
```

Before inputting to the conditional random field, we modify the fused saliency map as the following operation.

$$\beta_p(x_p) = -\frac{\log \hat{M}_p}{\tau\sigma\left(x_p\right)} \tag{11}$$

where $\hat{M}_p$ is the normalized value of each pixel x_p, $\sigma(\cdot)$ denotes the sigmoid activation function, and τ represents a scale factor. The $\beta_{pq}(x_p, x_q)$ is defined as

$$\beta_{pq}(x_p, x_q) = \mu\left(x_p, x_q\right)\left[\lambda_1 \exp\left(-\frac{\left\|\phi_p - \phi_q\right\|^2}{2\sigma_1^2} - \frac{\left\|V_p - V_q\right\|^2}{2\sigma_2^2}\right) + \lambda_2 \exp\left(-\frac{\left\|\phi_p - \phi_q\right\|^2}{2\sigma_3^2}\right)\right] \tag{12}$$

where $\mu(x_p, x_q) = 0$ if $x_p = x_q$, otherwise, $\mu(x_p, x_q) = 1$. ϕ_p and V_p respectively sands for the position and pixel value of x_p. λ_1 , λ_2 , σ_1 , σ_2 and σ_3 are the parameters of controlling the importance of the Gaussian kernel. We leverage the public tool, PyDenseCRF [35], to implement it. Here, because there are only two classes to segment, we directly treat the computed posterior probability of a pixel being the finer saliency map.

3.7. Inferential Process

To attain high model localization accuracy, the generation of a bounding box is needed. We input the image into the trained model to generate a fused

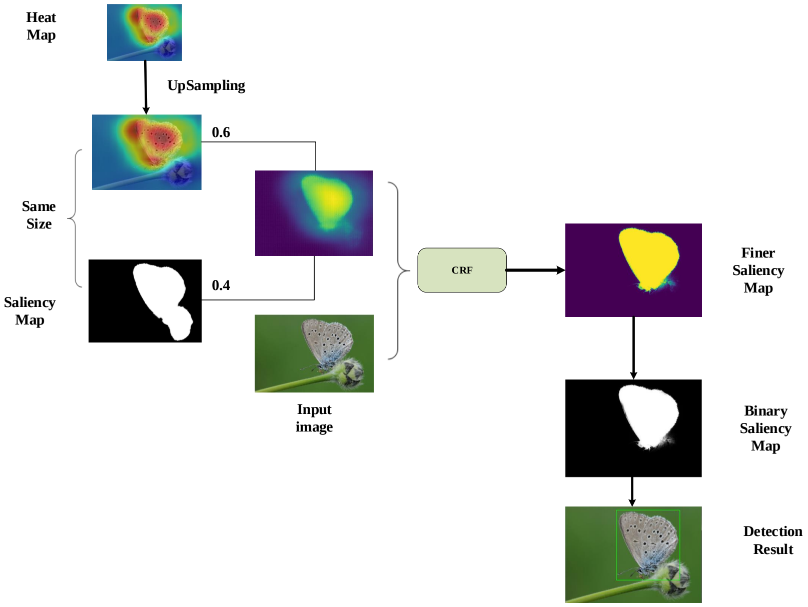


Figure 6: An example of generating the rectangular box.

SM. Then, a threshold is set to binarize the SM. Here, we choose 80% of the maximum pixel value as the threshold to segment the SM. In the end, the bounding box is set as a rectangular box covering the largest connected region. Figure 6 displays the procedure of generating the rectangular box.

4. Experimental results

To verify the performance of our proposed WBD-SM in butterfly object detection, we create an ecological image dataset containing twenty types of butterflies. Then, the recognition accuracy of our WBD-SM is compared with that of the VGG16 [5]. Next, we compare the localization accuracy of the WBD-SM with those of the DSS [6], and ACoL [7]. Then, to demonstrate that our weakly-supervised object detection model can achieve a competitive result with the fully-supervised object detection models, the detection accuracy of the WBD-SM is compared with those of the RefineDet [8], YOLOv3 [9], and SSD [10]. Finally, we investigate the effects of the CAM acquisition method and threshold erasing on the achievable performance of our WBD-SM. All experiments in this paper are implemented using the open-source, deep learning framework PyTorch. The experimental platform is an Nvidia Tesla K40c GPU server, and the memory size during training is 16 GB. The server CPU model is Intel® Xeon® E5-2643, while the operating system is Windows 7.

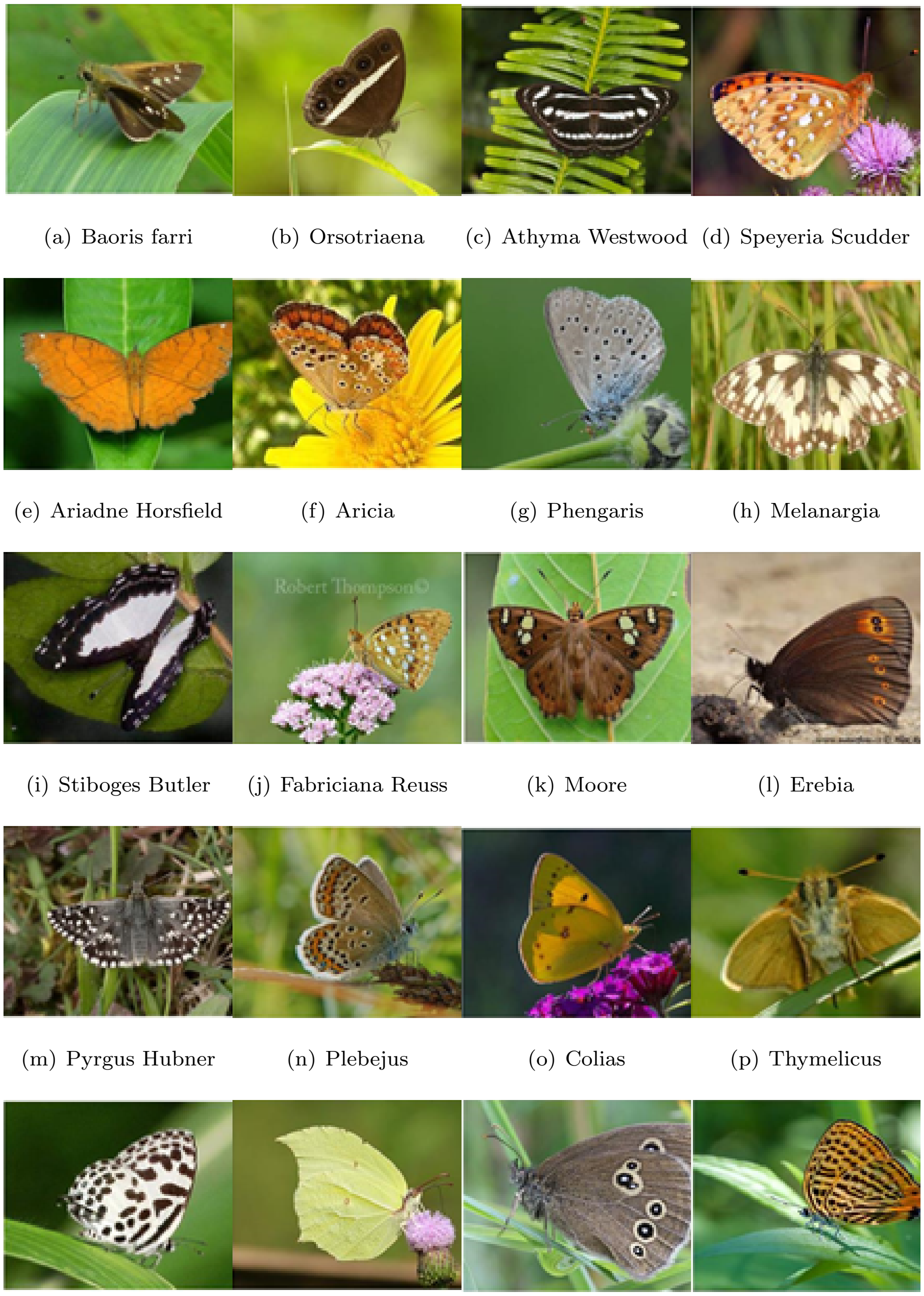
4.1. Dataset

Through field photography and web crawlers, a butterfly object detection dataset, "Butterfly20", is created that contains twenty genera of butterflies. In Figure 3, the example images of these twenty butterfly types are illustrated. For each genus, the number of images is 101 or 102, and there is a total of $2,026$ butterfly images. The range of the means of twenty classes is from 0.2484 ± 0.4061 to 0.5306 ± 0.4593, and the Pearson correlation coefficient between different classes is from 0.9501 to 0.9972. The dataset is divided into a training set and a test set with two ratios for each class: 8:2 and 7:3. For the 8:2 ratio, the training set contains $1,621$ images, whereas the test set includes 405 images. For the 7:3 ratio, the training set contains $1,418$ images, whereas the test set includes 608 images.

To perform the saliency detection task, the trained DSS is utilized to generate the butterfly image's rough saliency label. The saliency labels are normalized to within [0, 1] during the network input. To improve the model's detection capability, the training set is augmented during training, which includes the horizontal flip, vertical flip, and random alteration of image brightness, contrast, and saturation.

4.2. Parameter Setting and Evaluation Indices

The WBD-SM is trained with the training Algorithm 1. Before the training, ImageNet [36] is used to pre-train the convolutional part of the initial VGG16. The input image size is set to 256×256, and each mini-batch contains 16 images. The learning rate is an exponentially decaying learning rate, whose initial value is set to 0.0001, with a decay rate of 0.96. The Adam optimizer [37] is used, and a total of 25 epochs are iterated.



(a) Baoris farri (b) Orsotriaena (c) Athyma Westwood (d) Speyeria Scudder

(e) Ariadne Horsfield (f) Aricia (g) Phengaris (h) Melanargia

(i) Stiboges Butler (j) Fabriciana Reuss (k) Moore (l) Erebia

(m) Pyrgus Hubner (n) Plebejus (o) Colias (p) Thymelicus

(q) Castalius (r) Gonepteryx (s) Aphantopus (t) Japonica

Figure 7: Images of the 20 butterflies

The model performance is evaluated from two perspectives: the recognition accuracy and the location accuracy. Since our method is weakly supervised learning, we realized the butterfly detection with only image-level labels. Therefore, the location accuracy is relatively more important than the recognition accuracy here.

For the recognition accuracy, the top1 classification accuracy is adopted, which is defined as the fraction of the test images for which the top class label predicted (the one having the highest probability) is the same as the correct label. On the other hand, the location accuracy (Loc_Acc) is evaluated with the frame per second (FPS) and the intersection over union (IoU). The FPS refers to the number of images processed per second. The larger the FPS is, the faster the model is running. The IoU is the area of intersection between computationally predicted and labeled bounding boxes divided by the area of their union.

IoU means the area of intersection between predicted and ground-truth bounding boxes divided by the area of their union. It is computed as:

$$IoU = \frac{area(P) \cap area(G)}{area(P) \cup area(G)}, \tag{13}$$

where P and G stand for the predicate and ground-truth bounding box, respectively. Figure 8 shows its computing style.

We choose 0.5 as the threshold for several object detection methods, for example, MDFN [38], Gated CNN [39], and STDnet-ST [40] . If the value of IoU is more than 0.5, we treat it to locate accurately; otherwise, locating inaccurately. If the threshold is less than 0.5, the location accuracy will rise. However, there will appear some inaccurate locations, even wrong locations. If the threshold is greater than 0.5, the location accuracy will drop. Similarly, the location boxes will be more accurate.

4.3. Comparison of Recognition Results

To verify the effectiveness of the saliency detection in the WBD-SM network, the WBD-SM recognition results are compared with the VGG16 recognition

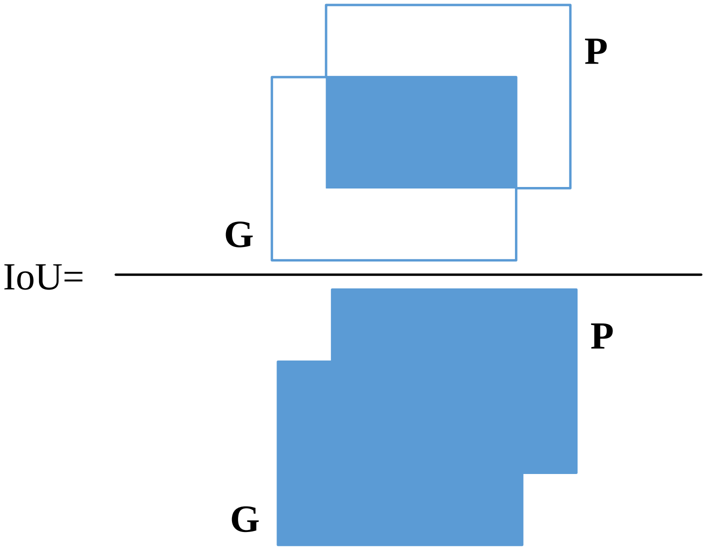
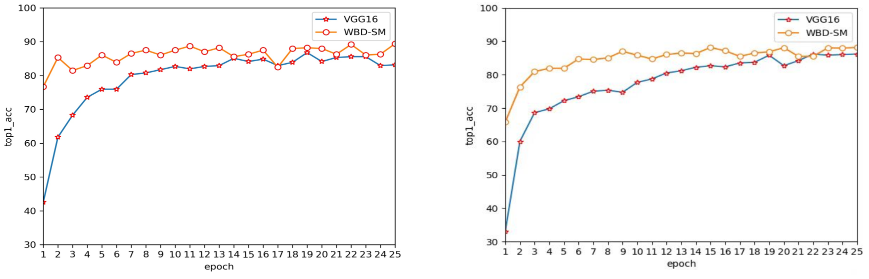


Figure 8: Computing style of the IoU.



(a) 8:2 division ratio

(b) 7:3 division ratio

Figure 9: Comparison of the recognition accuracies attained by WBD-SM and VGG16 with different dividing ratios.

results. For a fair comparison, the same optimizer, initial learning rate, decay rate, and number of iterations are adopted for the two networks. Figure 9 compares the recognition accuracies attained by the WBD-SM and VGG16 with different splitting ratios.

As shown in Figure 9, we can get that:

(1) For the 8:2 division ratio, the recognition accuracy of the WBD-SM is always higher than the VGG16 at the same number of iterations, except for epoch 17. After convergence, the recognition accuracy of the WBD-SM reaches 89.4%, which is 2.6% higher than 86.8% achieved by the VGG16 on the test data. Particular noteworthy is that the accuracy of the WBD-SM already reaches 76.7% after the first epoch, while the accuracy of the VGG16 is a mere 42.5% after the first epoch.

(2) For the 7:3 division ratio, the recognition accuracy of WBD-SM shows the same trend as 8:2 division ratio. After convergence, the recognition accuracy of WBD-SM achieves 88.16%, which increases by 1.95% than 86.19% got by VGG16 on the test data.

(3) This suggests that due to the integration of the saliency detection task in the WBD-SM, the model can achieve faster localization of areas conducive to recognition, which also yields a slightly improved final acceptance accuracy.

4.4. Comparison of Target Location Results

To verify the superior localization performance of the WBD-SM, Table 3 compares the butterfly localization accuracy attained by our WBD-SM with

Table 3: Localization accuracy results of three models with different splitting ratios

Methods	Loc_Acc(8:2) (%)	Loc_Acc (7:3) (%)
DSS [6]	86.30	83.78
ACoL [7]	83.89	77.01
WBD-SM	**95.67**	**92.76**

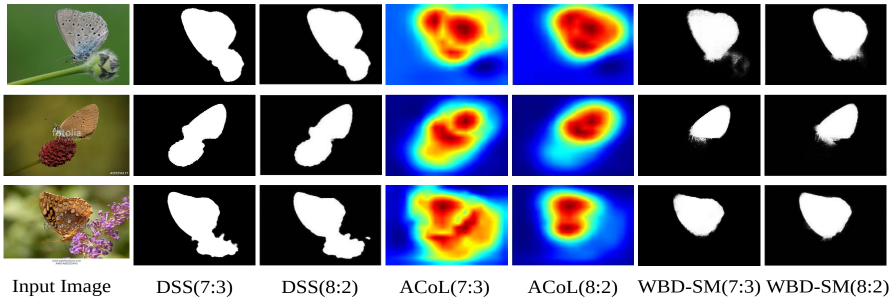


Figure 10: Comparison of three models' output results with different dividing ratios.

those achieved by the DSS and ACoL, while Figure 10 illustrates the output results from the three models.

According to the results of Table 3 and Figure 10, we can draw the following observations:

(1) For the 8:2 division ratio, the WBD-SM attains a 95.67% localization accuracy, which increases by 9.37% and 11.87% over the DSS (86.30%) and ACoL (83.89%), respectively. For the 7:3 division ratio, the WBD-SM attains a 92.76% localization accuracy, which increases by 8.98% and 15.75% over the DSS (83.78%) and ACoL (77.01%), respectively. This demonstrates the superior localization ability of our proposed WBD-SM over well-established models.

(2) The WBD-SM highlights the locations of butterfly objects by fusing the results of its two component networks. Consequently, unlike the other two models, it can display butterfly objects but not the rest of the objects that occupy salient locations.

Table 4: Comparison of WBD-SM with fully-supervised detection models

Models	Loc_Acc (%)	Speed (FPS)
RefineDet [8]	94.02	7.300
SSD [10]	**95.69**	8.403
YOLOv3 [9]	91.10	9.132
WBD-SM	95.67	**14.345**

4.5. Detection Performance Comparison Between WBD-SM and Fully-supervised Object Detection Networks

To verify the detection performance of the WBD-SM, we further compare it with the fully-supervised RefineDet, SSD and YOLOv3. In Table 4, the localization results of these four models under the 8:2 division ratio are compared. It can be seen that the WBD-SM attains a 95.67% localization accuracy, which is 1.65% and 4.57% higher than the results obtained by the RefineDet and YOLOv3, respectively, while it is only 0.02% lower than the result of the SSD. Besides, our model got the speed of 14.345 FPS, higher than that all of other models. This indicates that by combining SMs with adversarial erasing, the weakly supervised WBD-SM can yield a competitive result with fully supervised state-of-the-art object detection models.

4.6. Effect of CAM Acquisition Method on the Localization Performance

To obtain accurate CAMs, the efficiency of CAMs generated by the top 5 prediction classes is first analyzed. In Figure 11, the CAMs generated by the top 5 prediction classes under the 8:2 division ratio are compared are presented. It is clear that the CAM generated by the higher-ranking prediction class displays the target location better and is more reliable. Although the top1 CAM has the highest reliability, the CAMs generated by other prediction classes may discover the parts that the top1 CAM misses. Hence, they can serve as a supplement to the top1 CAM.

Next, we investigate the effects of various CAM acquisition methods on the localization results by utilizing only CAM localization. In Table 4, four types of CAM acquisition methods are compared, where 'top1' represents the CAM generated by the first prediction class only, and 'top5(0.5)' represents the mean fusion of the CAMs generated by the top 5 prediction classes, while 'top3(0.3)' represents the mean fusion of the CAMs generated by the top 3 prediction classes, and 'top3(3:2:1)' represents the weighted fusion of the top 3 classes according to a weight ratio of 3:2:1 (0.57:0.28:0.14). From Table 5, we can draw the following observations.

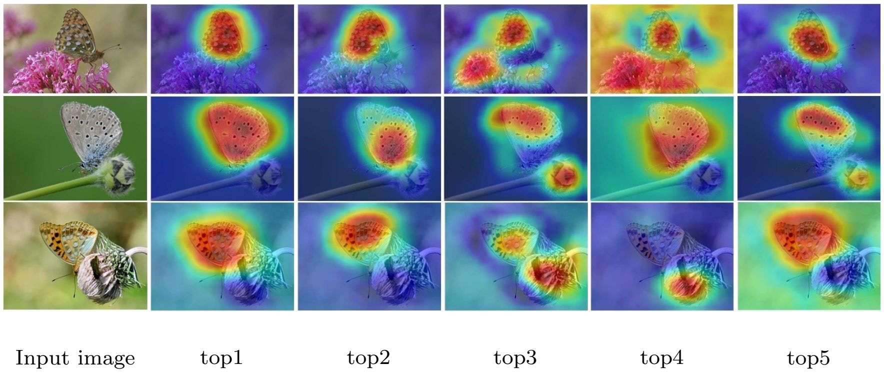


Figure 11: Top1 to top5 class activation maps

(1) As suggested by the results of Experiments 1 and 2, the fusion with too low-rank CAMs leads to the reduction in localization accuracy. This is because the lower the prediction class's ranking, the less its reliability. Low-rank CAMs leads to the reduction in localization accuracy. This is because the low-rank CAMs stand for the wrong class label of the butterfly image, and therefore its CAM is wrong, focusing on different areas. This will provide the wrong information for the saliency map, leading to the reduction in localization accuracy.

(2) Comparison between Experiments 1 and 3 reveals that the fusion of the CAMs up to the top 3 classes helps to enhance the localization accuracy. This is because apart from the best top1 class, the effective localization regions are also found in the CAMs of top2 and top3 classes.

(3) Experiment 4 attains the highest localization accuracy by fusing the top 3 classes with higher-rank class principles having higher weight. The weight ratio 3:2:1 (0.57:0.28:0.14) is found empirically.

4.7. Effect of Erasing Threshold on the Localization Performance

During the generation of CAMs, the two adversarial branches, A and B, learn different regions of interest. Branch A learns the most discriminative area, whereas branch B learns the second discriminative region. The most discriminative region is determined by the threshold ρ. A minimal threshold value

Table 5: Localization accuracy of different CAMs

Experiment number	CAMs	Location accuracy (%)
1	top1	78.13
2	top5 (0.5)	75.00
3	top3 (0.3)	83.89
4	top3 (3:2:1)	**87.26**

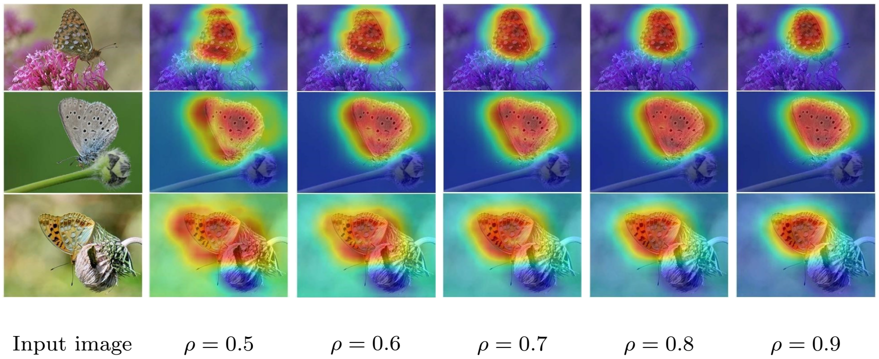


Figure 12: Effect of erasing threshold ρ on class activation maps

makes the network discover additional regions, thereby bringing in noise. By contrast, an overly large threshold prevents the network from learning sufficient regions. Therefore, an appropriate threshold is essential for the two branches to jointly learn a proper range of regions.

In Figure 12, the efficiencies of the generated CAMs are displayed at various thresholds under the 8:2 division ratio. As can be seen, when $\rho = 0.5$, the red area also covers the regions other than the butterflies, which presents disconnected sites. With the increase of ρ, the red area is more connected, but its coverage area decreases.

In Table 6, the effects of various erasing thresholds on the localization accuracy are investigated. In Experiments 5-9, the SMs are obtained by fusing the CAMs with the SMs after three iterations (SMP3). In Experiments 10-14, the SMs are accepted only from the CAMs. Both sets of experiments demonstrate that the highest localization accuracy is attainable at the threshold value of $\rho = 0.8$. At thresholds lower or higher than this value, the localization accuracy

Table 6: Effect of Erasing Threshold on Localization Accuracy

Experiment number	ρ	Saliency Map	Location accuracy (%)
5	0.5	CAMs+SMP3	94.47
6	0.6	CAMs+SMP3	94.71
7	0.7	CAMs+SMP3	95.43
8	0.8	CAMs+SMP3	**95.67**
9	0.9	CAMs+SMP3	94.47
10	0.5	CAMs	56.85
11	0.6	CAMs	68.99
12	0.7	CAMs	78.13
13	0.8	CAMs	**87.98**
14	0.9	CAMs	84.13

Table 7: Results of WBD-SM with different backbone networks

Models	Location accuracy(%)	Speed(FPS)
WBD-SM-Inception	92.60	9.132
WBD-SM-ResNet50	93.65	9.674
WBD-SM-DenseNet121	94.40	7.641
WBD-SM-MobileNet128	95.21	10.239
WBD-SM-VGG16	**95.67**	**14.345**

all degrades to some extents. These findings are consistent with the results visualized in Figure 7.

4.8. Effect of Backbone Networks on the Localization Performance

To investigate the backbone network on the performance of WBD-SM, the location accuracy and speed are obtained from WBD-SM with different backbone networks are analyzed. We equipped our model with different backbone networks. They are Inception, ResNet50, DenseNet121, and MobileNet28 [**?**], and the corresponding models are represented as WBD-SM-Inception, WBD-SM-ResNet50, WBD-SM-DenseNet121, WBD-SM-MobileNet128, and WBD-SM-VGG16, respectively. Table 7 gives their results.

From Table 7, we can get that WBD-SM-VGG16 gets the 95.67% location accuracy, higher than that of all the other models. This indicates that VGG16 as the backbone network can combine the abstract and low-level features to provide suitable information for the saliency map. Additionally, we can obtain the speed of 14.345 FPS, faster than that all of the other models.

4.9. Comparison on Alternate Dataset

To validate our method on other datasets, we compared our method with RefinDet, SSD and YOLOv3 on the Oxford-IIIT Pet Dataset, which is available online, i.e., [www.robots.ox.ac.uk/ vgg/data/pets/]. The Oxford-IIIT Pet dataset has 37 categories with roughly 200 images for each class. The images

Table 8: Results of four models on the Oxford-IIIT Pet dataset.

Models	Location Accuracy (%)	Speed (FPS)
RefineDet	90.90	4.493
SSD	90.80	6.190
YOLOv3	**94.80**	**7.857**
WBD-SM	93.88	5.213

have significant variations in scale, pose and lighting. All images have an associated ground truth annotation of bread, head ROI, and pixel-level trimap segmentation. Table 8 listed the results of these four models on this dataset. From Table 8, we can get that WBD-SM achieves the location accuracy of 93.88%, higher than that of both RefineDet and SSD models. Specifically, it improves by 2.78% and 3.08%, respectively. When comparing with YOLOv3, it is lower by 0.92% than that of YOLOv3. Secondly, the speed of WBD-SM is 5.213 FPS, faster than both RefineDet and SSD, while slower than that of YOLOv3. This demonstrates that our WBD-SM model can achieve competitive results of both location accuracy and speed on a larger dataset than fully supervised models.

5. Conclusions and Future Research

This paper has proposed a weakly supervised butterfly detection based on a saliency map (WBD-SM). Our WBD-SM uses VGG16 without fully connected layers to extract features of different scales, which serves as the backbone network. DSS is utilized to remove the SMs of butterfly images, and the CAMs of butterfly images are derived via the ACoL network. Afterwards, the SMs and CAMs are post-processed with conditional random fields, thereby obtaining the refined SMs of butterfly objects. Finally, the locations of the butterflies are acquired based on the SMs. The experimental results involving a butterfly dataset with 20 categories of butterfly have demonstrated that the proposed WBD-SM considerably outperforms DSS and ACoL, in terms of localization accuracy, while our WBD-SM achieves a higher recognition accuracy than that of VGG16. The experiments have also shown that our weakly-supervised WBD-SM yields competitive results with fully supervised state-of-the-art object detection models, including RefineDet, YOLOv3 and SSD, in terms of detection performance.

Although the WBD-SM has achieved favourable results, many aspects of this novel model require further studied. Given the inability of saliency detection to implement semantic and instance discrimination, the detection network proposed in the WBD-SM can only detect a single butterfly object in the images. It is incapable of achieving a simultaneous distinction between multiple butterfly objects. Although this paper has obtained excellent localization results by fusing SMs with CAMs, multiple repeated steps are needed to yield accurate results. Additionally, the two subtasks remain highly independent of each other, and they are not integrated deeply during network training. In future research, the network needs to be made adaptable to various scenarios. The association between the two tasks should be explored further to develop a better way of integrating them. There is also a need to collect bigger and comprehensive butterfly datasets.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (61806013, 61876010,61906005), General project of Science and Technology Plan of Beijing Municipal Education Commission (KM202110005028), Project of Interdisciplinary Research Institute of Beijing University of Technology (2021020101) and International Research Cooperation Seed Fund of Beijing University of Technology (2021A01).

References

[1] O. E. Sala, F. S. Chapin, J. J. Armesto, E. Berlow, *et al.*, "Global biodiversity scenarios for the year 2100," *Science*, vol. 287, no. 5459, pp. 1770–1774, 2000.

[2] K. J. Gaston and M. A. O'Neill, "Automated species identification: why not?," *Phil. Trans. Royal Society B: Biological Sciences*, vol. 359, no. 1444, pp. 655–667, 2004.

[3] M. Espeland, J. Breinholt, K. R. Willmott, A. D. Warren, *et al.*, "A comprehensive and dated phylogenomic analysis of butterflies," *Current Biology*, vol. 28, no. 5, pp. 770–778, 2018.

[4] S. U. Rehman, S. Tu, M. Waqas, Y. Huang, O. U. Rehman, B. Ahmad, and S. Ahmad, “Unsupervised pre-trained filter learning approach for efficient convolution neural network,” in *Neurocomputing*, vol. 365, pp. 171-190, Nov. 2019.

[5] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Proc. ICLR 2015* (San Diego, CA, USA), May 7-9, 2015, pp. 1–14.

[6] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, *et al.*, “Deeply supervised salient object detection with short connections,” in *Proc. CVPR 2017* (Honolulu, HI, USA), Jul. 21-26, 2017, pp. 3203–3212.

[7] X. Zhang, Y. Wei, J. Feng, Y. Yang, *et al.*, “Adversarial complementary learning for weakly supervised object localization,” in *Proc. CVPR 2018* (Salt Lake City, UT, USA), Jun. 18-22, 2018, pp. 1325–1334.

[8] S. Zhang, L. Wen, X. Bian, Z. Lei, *et al.*, “Single-shot refinement neural network for object detection,” in *Proc. CVPR 2018* (Salt Lake City, UT, USA), Jun. 18-22, 2018, pp. 4203–4212.

[9] J. Redmon and A. Farhadi, “YOLOv3: an incremental improvement,” *arXiv:1804.02767*, pp. 1–6, 2018.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, *et al.*, “SSD: single shot multibox detector,” in *Proc. ECCV 2016* (Amsterdam, Netherlands), Oct. 11-14, 2016, pp. 21–37.

[11] Z. Liu, M. Waqas, J. Yang, A. Rashid and Z. Han, “A Multi-Task CNN for Maritime Target Detection,” in *IEEE Signal Processing Letters*, vol. 28, pp. 434-438, 2021.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. NIPS 2012* (Lake Tahoe, NV, USA), Dec. 3-8, 2012, pp. 1097–1105.

[13] S. Tu, S. U. Rehman, M. Waqas *et al.*, "ModPSO-CNN: an evolutionary convolution neural network with application to visual recognition," in *Soft Computing*, vol. 25, pp. 2165-2176, Sept. 2020.

[14] R. Girshick, "Fast R-CNN," in *Proc. ICCV 2015* (Santiago, Chile), Dec. 11-18,, 2015, pp. 1440–1448.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *Proc. NIPS 2015* (Montreal, Quebec, Canada), Dec. 7-12, 2015, pp. 91–99.

[16] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proc. ICCV 2017* (Venice, Italy), Oct. 22-29, 2017, pp. 2980–2988.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proc. CVPR 2016* (Las Vegas, NV, USA), Jun. 26-Jul. 1, 2016, pp. 779–788.

[18] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proc. CVPR 2017* (Honolulu, HI, USA), Jul. 21-26, 2017, pp. 6517–6525.

[19] S. Zhang, L. Wen, X. Bian, Z. Lei, *et al.*, "Single-shot refinement neural network for object detection," in *Proc. CVPR 2018* (Salt Lake City, UT, USA), Jun. 18-22, 2018, pp. 4203–4212.

[20] X. Li, M. Kan, S. Shan, and X. Chen, "Weakly supervised object detection with segmentation collaboration," in *Proc. ICCV 2019* (Seoul, South Korea), Oct. 27-Nov. 2, 2019, pp. 9735–9744.

[21] R. G. Cinbis, J. Verbeek, and C. Schmid, "Weakly supervised object localization with multi-fold multiple instance learning," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 39, no. 1, pp. 189–203, 2017.

[22] H. Bilen and A. Vedaldi, "Weakly supervised deep detection networks," in *Proc. CVPR 2016* (Las Vegas, NV, USA), Jun. 26-Jul. 1, 2016, pp. 2846–2854.

[23] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, *et al.*, “Learning deep features for discriminative localization,” in *Proc. CVPR 2016* (Las Vegas, NV, USA), Jun. 26-Jul. 1, 2016, pp. 2921–2929.

[24] Q. Wang, L. Zhang, Y. L, and K. Kpalma, “Overview of deep-learning based methods for salient object detection in videos,” *Pattern Recognition*, vol. 104, pp. 1–16, 2020.

[25] W. Wang, J.Shen, and L. Shao, “Video salient object detection via fully convolutional networks,” *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 38–49, 2018.

[26] M. Feng, H. Lu, and E. Ding, “Attentive feedback network for boundary-aware salient object detection,” in *Proc. CVPR 2019* (Long Beach, CA, USA), Jun. 16-20, 2019, pp. 1623–1632.

[27] C. Caelles, K.-K. Maninis K, J. Pont-Tuset, L. Leal-Taixe, *et al.*, “One-shot video object segmentation,” in *Proc. CVPR 2017* (Honolulu, HI, USA), Jul. 21-26, 2017, pp. 5320–5329.

[28] Z. Liu, Z. Zhang, T. Jiang, T. Zhang, B. Liu, M. Waqas, and Y. Li, ”Infrared salient object detection based on global guided lightweight non-local deep features,” in *Infrared Physics & Technology*, vol. 115, pp. 103672, Jun. 2021.

[29] B. Pan, X. Xu, Z. Shi, N. Zhang *et al.*, “DSSNet: a simple dilated semantic segmentation network for hyperspectral imagery classification,” *IEEE Geoscience and Remote Sensing Letters*, pp. 1–5, 2020.

[30] M. Sulaiman, Z. Halim, M. Waqas *et al.*, “A hybrid list-based task scheduling scheme for heterogeneous computing,” *The Journal of Supercomputing*, pp. 1-37, Mar. 2021.

[31] M. Sandler, A. Howard, M. Zhu, *et al.* ”MobileNetV2: Inverted Residuals and Linear Bottlenecks,” *Proc. CVPR 2018*, (Salt Lake City, Utah), Jun. 19-21, 2018, pp. 4700-4708.

[32] Y. Ho and S. Wookey, “The real-world-weight cross-entropy loss function: modeling the costs of mislabeling,” *IEEE Access*, vol. 8, pp. 4806–4813, 2020.

[33] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, “A tutorial on the cross-entropy method,” *Annals Operations Research*, vol. 134, pp. 19–67, 2001.

[34] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” in *Proc. CVPR 2018* (Salt Lake City, UT, USA), Jun. 18-22, 2018, pp. 7482–7491.

[35] P. Krahenbuhl, V. Koltun., ”Efficient inference in fully connected crfs with gaussian edge potentials,” in *Proc. NIPS 2011* (Granada Spain), Dec. 12-17, 2011, pp. 109–117.

[36] Stanford Vision Laboratory: ImageNet. http://www.image-net.org/.

[37] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980.*

[38] W. Ma, Y. Wu, F. Cen *et al*, “MDFN: Multi-scale deep feature learning network for object detection,” *Pattern Recognition*, vol. 100, no. 107149, 2020.

[39] J. Yuan, H. C. Xiong, Y. Xiao, W. Guan, M. Wang, R. Hong, Z. Y. Li, “Gated CNN: Integrating multi-scale feature layers for object detection,” *Pattern Recognition*, vol. 105, no. 107131, 2020.

[40] B. Bosquet, M. Mucientes, V. M. Brea, “STDnet-ST: Spatio-temporal ConvNet for small object detection,” *Pattern Recognition*, vol. 116, no. 107929, 2021.