Weakly Supervised Posture Mining for Fine-grained Classification (Supplementary Material)

Zhenchao Tang[1,†], Hualin Yang[1,†], and Calvin Yu-Chian Chen[1,2,3,*]
[1]Sun Yat-sen University, [2]China Medical University Hospital, [3]Asia University
tangzhch7@mail2.sysu.edu.cn, yanghlin8@mail2.sysu.edu.cn, chenyuchian@mail.sysu.edu.cn

1. OverView

In this supplementary material, we present more experimental details and results. Experimental details include: section2-5, more experimental results include: section6 and section7.

2. Detailed setting of Deep Navigator

Deep Navigator obtains feature maps on three scales: $\{14\times14, 7\times7, 4\times4\}$. We set a certain number of anchors on the large-scale feature maps and more anchors on the small-scale feature maps, so as to balance the number of anchors at different scales. The number of anchors is {6,6,9} respectively (To exmine more tiny discriminative regions of the objects, we use the same number of anchors in the size of $\{14\times14\}$ and the size of $\{7\times7\}$). We do not need to correct the anchor position, so we do not need ground truth as a monitoring signal. We only need to obtain the informativeness of each anchor. And the informativeness reflects the probability that the anchor has discriminative regions. Then we use Soft-NMS [1] to filter out the discriminative regions we need according to the informativeness. The architecture of the Deep Navigator is shown in Figure. 1.

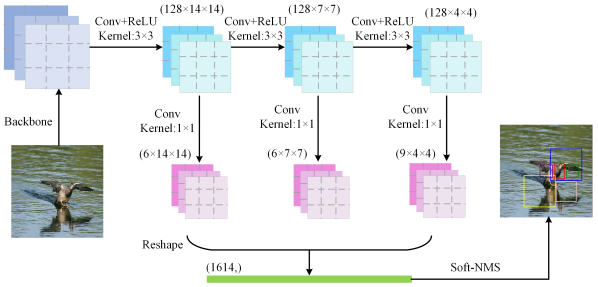

Figure 1. The architecture of the Deep Navigator. The Deep Navigator can generate multi-scales discriminative regions. We set the amount of the anchors as {6,6,9} respectively on the feature maps of three scales.

[†]Equal contribution. [*]Corresponding author.

3. Details of Graph construction

For region r_m and r_n, we set the center coordinates of the two regions in the original image as (x_m, y_m) and (x_n, y_n), and the edge weight between the two regions is:

$$W_{m,n} = W_{n,m} = \sigma^2 e^{-\frac{\sqrt{(x_m-x_n)^2+(y_m-y_n)^2}}{2l^2}} \quad (1)$$

where σ and l are used to balance the order of magnitude between the weight value of the edge and feature value of the node. The closer the two nodes are, the greater their edge weight value, which means that the relationship between the two nodes is closer. Besides, we set the parameters σ and l as learnable parameters.

For the features of nodes, we first crop and resize the discriminative regions from the original image to a unified size, and use the same feature extractor to extract the high-level semantic features of the discriminative regions. The feature extractor is the backbone and share parameters. The feature extractor abstracts the discriminative regions into a multi-channel feature map. We use global average pooling operation to convert the feature map into a feature vector, which will be added to the graph as the node feature corresponding to the discriminative regions. Considering the inference speed, we can directly use bilinear interpolation sampling on the feature maps of the backbone to obtain the features of the discriminant region, and compress the features of the discriminant region into vectors through global average pooling.

4. Training steps

We record the parameters of the backbone as $W_{backbone}$, the parameters of the Deep Navigator as $W_{navigator}$. The training process of this method is shown in Algorithm 1.

5. Data preprocessing and Hyperparameter setting

In our experiment, all the input images were preprocessed to 600 × 600. We randomly rotate the image within

Algorithm 1 Training our method

Input : Training dataset $\{image_i, label_i\}_{i=1}^{N}$, hyper-parameters $M, \alpha, \beta, \gamma, \theta$.
Output : The score of each category.

For $t = 1 \cdots T$ **do**
  For $i = 1 \cdots N$ **do**
    Get a sample $iamge_i$ as x, get the feature: raw=$backbone(x)$.
    Generate anchors: $\{A_1, A_2, \cdots, A_{1614}\}$.
    Calculate the information: $\{I'_i\}_{i=1}^{1614}$=$Navigator(backbone(x))$.
    Get the discriminative regions: $\{I_i\}_{i=1}^{M}, \{R_i\}_{i=1}^{M}$=Soft-NMS$(\{I'_i\}_{i=1}^{1614}, \{A_i\}_{i=1}^{1614})$.
    Construct the graph G with the coordinate of the discriminative regions from Supplementary Eq.(1), add the node feature: $\{h_i\}_{i=1}^{M}$=$backbone(\{R_i\}_{i=1}^{M})$.
    Message Passing on the graph G to update the node feature as $\{h\}_{i=1}^{M}$ from Submission Eq.(1)-(3), calculate the confidence of the node as $\{C_i\}_{i=1}^{M}$, calculate the score of the node from Submission Eq.(6).
    Readout the graph feature as $score$ from Submission Eq.(4).
    $BP(L)$ get the gradient of: $\{W_{backbone}, W_1, W_2, W_{navigator}\}$.
    Update $\{W_{backbone}, W_{navigator}, W_1, W_2\}$ using SGD.
end

a 45 degree angle, and cut the area size of $\{448 \times 448\}$ from the images, and the images are standardized on three channels, with mean value (0.485, 0.456, 0.406) and standard deviation (0.229, 0.224, 0.225). We use ResNet-50 [2] as the default backbone and use the pre-training model on Imagenet [3] to initialize ResNet-50. We use Momentum SGD with initial learning rate 1e-4 and the learning rate is multiplied by 0.1 after 40 epochs, and we use weight decay 1e-4. The Soft-NMS threshold is set to 0.25. The proposed method is trained using 600 epochs with a batch size of 8. All of our experiments are conducted on PyTorch with Nvidia Tesla V100 GPUs.

6. Supplementary Results of Ablation Experiments

The number of discriminative regions affects the classification accuracy of the model, as shown in Figure. 2. As we can see from Figure. 2, when the number of the discriminative regions is small, the lack of some information leads to the low classification accuracy of the model. With the increase of the number of the discriminative regions, more information is introduced and the classification accuracy of the model is improved and stabilized within a certain range. In addition, an appropriate number of the discriminative regions can make full use of the message passing mechanism to improve Top-1 accuracy.

We test the proportion of $\alpha, \beta, \gamma, \theta$. We set six different propotions in Table 1. According to Submission Eq.(11), $\alpha, \beta, \gamma, \theta$ are the coefficients of $L_{backbone}, L_{navigator}, L_{message}, L_{graph}$ in total loss. The function of backbone is to extract features. So the performance of backbone usually affects the Deep Navigator and message passing. And the Deep Navigator and message passing can directly affect the recognition results of the model. We set $\alpha = \beta = \gamma = \theta = 0.25$ as the benchmark, when we only increase β to 0.4 or γ to 0.4, the accuracy will be significantly improved to 89.5% and 90.9% respectively. If we only increase α, the accuracy will drop from 88.7% to 88.2%. L_{graph} using RCE to calculate the loss and corresponds to the last layer classifier of the whole graph classification, and L_{graph} also directly affects the recognition accuracy of the model. Besides, the characteristic of RCE to increase the inter-class differences can further help the model improve the recognition accuracy. Therefore, when we increase β, γ, θ at the same time, the recognition accuracy of the model reaches the highest of 91.8%.

Table 1. Ablation Study of Loss Propotion on CUB-200-2011, backbone: ResNet50

α	β	γ	θ	Top-1 Accuracy(%)
0.25	0.25	0.25	0.25	88.7
0.2	0.2	0.2	0.4	91.2
0.4	0.2	0.2	0.2	88.2
0.2	0.4	0.2	0.2	89.5
0.2	0.2	0.4	0.2	90.9
0.1	0.25	0.25	0.35	**91.8**

7. Supplementary Results of RCE

In Figure. 3, we use t-SNE to visualize the distribution of the model output. It can be clearly seen that the clusters in the second row are far more apart than the clusters in the first row, which means that the performance of RCE is better than CE for fine-grained classification. Then, different columns represent the comprison of differnt models for fine-grained classification. The first two columns are the models not designed especially for fine-grained classification. We can see that the distribution of model output of these models are rather disordered which means that these models can not do well in fine-grained classification. And the third column represents the distribution of model output of NTS-Net, the

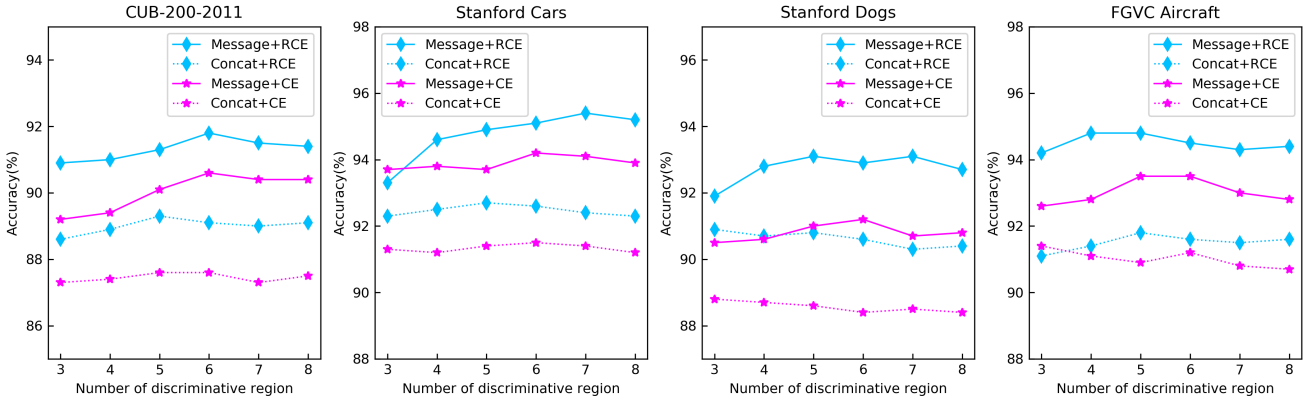

Figure 2. The relationship between accuracy and the number of discriminative regions. Under the condition of setting different number of discriminative regions, the model recognition effects of introducing message passing network and RCE training are tested respectively.

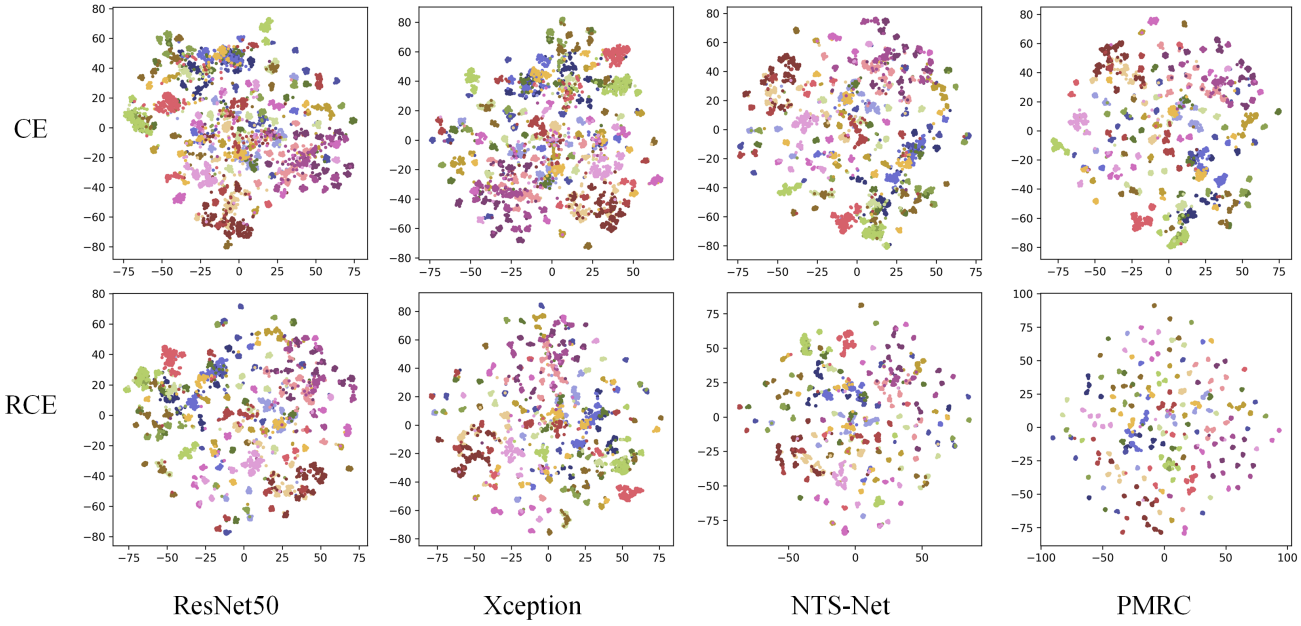

Figure 3. Based on t-SNE visualization, the training results of CE and RCE are compared. Different categories are marked with different colors.

clusters of which are more apart than the first two models. This means that NTS-Net is a better model for fine-grained classification than ResNet50 and Xception. It can be seen in the forth column that the clusters are far more apart than the first three columns which means that our model have better performance than these models and PMRC has the ability to classify the fine-grained samples well.

References

[1] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Improving object detection with one line of code. 2017. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[3] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 2