Supplemental Materials

Shichao Kan[1,2], Yigang Cen[1,2,*], Yang Li[3], Vladimir Mladenovic[4] and Zhihai He[3,*]

[1]Institute of Information Science, Beijing Jiaotong University, Beijing 100044, China

[2]Beijing Key Laboratory of Advanced Information Science and Network Technology, Beijing 100044, China

[3]Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO 65211, USA

[4]Faculty of Technical Sciences University of Kragujevac, Cacak, Serbia

16112062@bjtu.edu.cn; **ygcen@bjtu.edu.cn**; yltb5@mail.missouri.edu

vladimir.mladenovic@ftn.kg.ac.rs; **HeZhi@missouri.edu**

In this supplementary material, we provide the algorithm summary, more experimental results, and ablation studies for further understanding the proposed ROUL algorithm.

A. Summary of Algorithm

The proposed ROUL algorithm is summarized in Algorithm 1.

B. More Ablation Studies

In this section, we further analyze the contribution of each algorithm component of our method. We first generate the pseudo labels using the k-means clustering algorithm based on the ImageNet pre-trained model, then we conduct the following experiments with iterative pseudo labels generation.

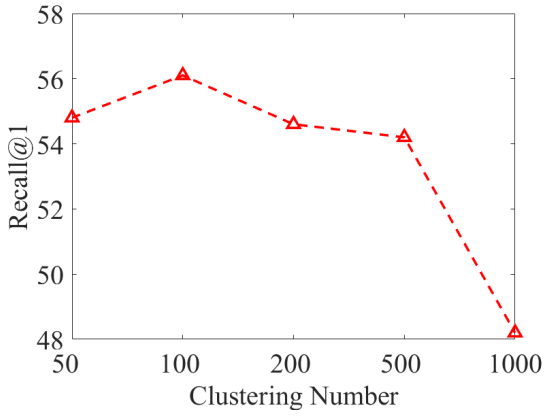

Figure 1. The impact of the clustering numbers.

(1) Impact of the clustering numbers. We evaluate our method with different numbers of clusters on the CUB dataset using the GoogleNet backbone. From Figure 1, we can see that our method achieves the best performance as the number of clusters approaches the number of ground truth classes (i.e., 100).

(2) Impact of the embedding size. Table 1 evaluates the impact of different embedding sizes on the CUB dataset with the GoogleNet backbone. We can see that the performance gradually improves with the dimension from 64 to

Algorithm 1 Summary of Optimization Algorithm

Initialization: Initialize parameters of backbone encoder with the ImageNet pre-trained model. Initialize the value of learning rate, epoch t, iterations k of each epoch.
Input: Training images.
Output: Optimized network parameter Θ.
for $iter_1$ in range(t) **do**
    Extract features of the training images.
    Calculate pseudo labels of the training images by the K-means clustering algorithm.
    Set the mini-batch size o as 100.
    for $iter_2$ in range(k) **do**
        Randomly sample a mini-batch images with corresponding pseudo labels from the training dataset.
        Do augmentation for each image in the mini-batch.
        Extract features of the sampled mini-batch images and the augmented images by the backbone encoder.
        for i in range(o) **do**
            Calculate the $\mathcal{L}_{ROC}$, $\mathcal{L}_{MOC}$, $\mathcal{L}_{FEN}$ and $\mathcal{L}_{O}$ losses.
        end for
        Calculate the overall loss.
        Back propagate and update parameters.
    end for
end for

* corresponding authors

512, and drops a little when the embedding size is increased to 1024.

Table 1. The retrieval performance with GoogleNet backbone for different embedding size on the CUB dataset.

Feature Dim	R@1	R@2	R@4	R@8
64	52.0	64.0	74.5	83.7
128	56.1	67.7	78.1	85.9
256	57.3	68.7	79.5	86.5
512	58.0	69.9	79.2	87.5
1024	57.2	69.0	78.6	86.5

C. Visualization of Learned Features

In order to visualize the learned feature, we use the t-SNE visualization method to show the results of the clustering on the CUB, Cars and SOP datasets, as shown in Figures 2, 3, and 4. Although it is hard to clearly see with a large number of classes being visualized together, from the enlarged samples, we can see that our algorithm is able to successfully aggregate images from the same classes in the high-dimensional feature space.

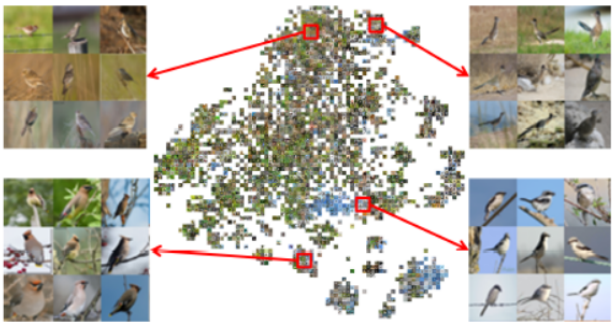

Figure 2. T-SNE visualization of the 128-dimensional embeddings with the GoogleNet backbone on the CUB test set.

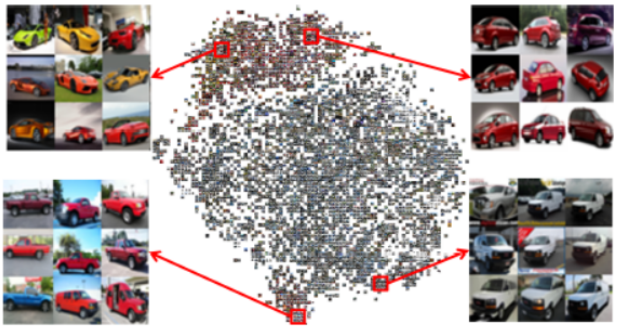

Figure 3. T-SNE visualization of the 128-dimensional embeddings with the GoogleNet backbone on the Cars test set.

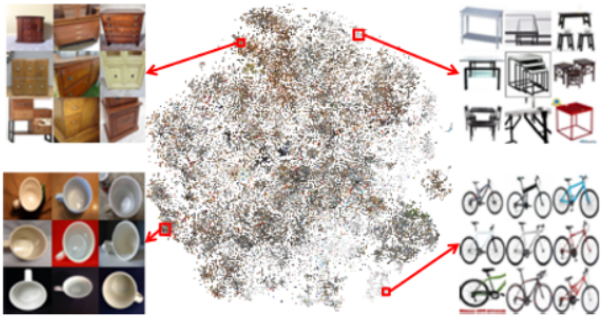

Figure 4. T-SNE visualization of the 128-dimensional embeddings with the GoogleNet backbone on the SOP test set.