CS4186 Assignment 1 Report – Research of Two Image Searching Methods

ZHANG Deheng 55199998

1. Introduction

This project aims to implement two algorithms for instance search. There is a database contains 5000 images, 10 ground truth query images and 20 query images. For each query images, the algorithm should output an ordered list of images with respect to the similarity between the query object and the object inside the database image. The algorithm is assessed by the mean average precision (MAP) which is defined as:

$$MAP = \frac{1}{m}\sum_{i=0}^{m-1}\frac{i+1}{S_i}$$

, where S is a sequence of ground truth image ranking. In this project, I implemented Convolutional Neural Network and a combined method based on Scale Invariant Feature Transform (SIFT) and Color Histogram (Color-hist). As a result, the CNN method has an acceptable performance (0.53 for the ground truth query) and faster computational time (with 1080ti GPU). However, the algorithm tends to classify the images on some query examples instead of detecting the objects since the model is based on pre-trained AlexNet, which is trained for classification problem. The implementation refers to [1]. For the combined method, the MAP is relatively low (0.42 for the ground truth query) and slower computation time. But the combination of SIFT and Color-hist provides a guideline to improve the performance of CNN. The implementation of this method is based on the *OpenCV* library [2].

2. Methodology

2.1. Method 1: CNN-based Method

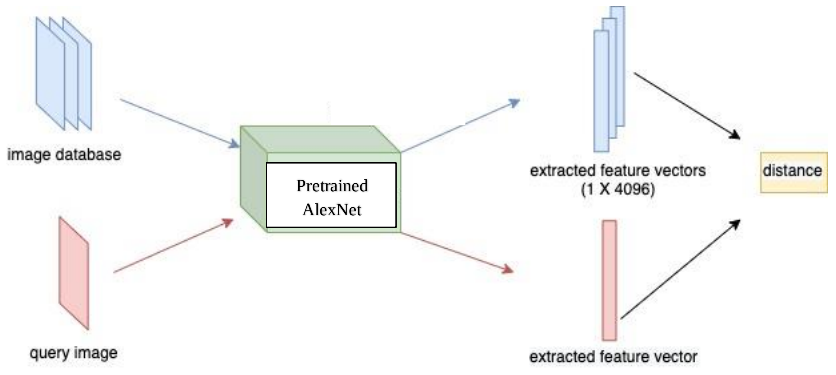

Figure 1: CNN Method Diagram

As shown in Figure 1, I used pre-trained AlexNet to extract the feature vectors from the database images. The feature is selected as the output before the average pooling since features after average pooling may loss information. During the image searching phase, the query image is transformed to the feature vector by using the same method. Then the algorithm computes the distance between the query feature vector and the set of database feature vectors. The distance between the query feature $\boldsymbol{q}$ and data feature $\boldsymbol{f}$ is defined as:

$$dist(\boldsymbol{q},\boldsymbol{f}) = \frac{\sum_{i=1}^{n}\boldsymbol{q}_i\boldsymbol{f}_i}{\sqrt{\sum_{i=1}^{n}\boldsymbol{q}_i^2}\sqrt{\sum_{i=1}^{n}\boldsymbol{f}_i^2}}.$$

The database images are sorted according to the distance in ascending order.

2.2. Method 2: Combined Method

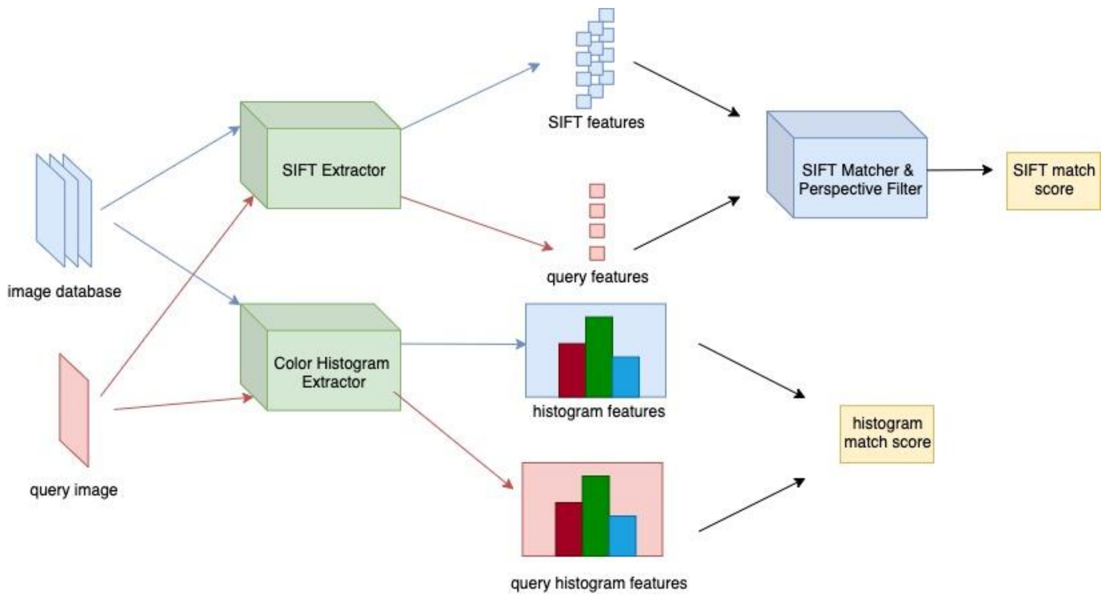

Figure 2: Combined Method Diagram

For the second method, I used SIFT and color histogram in RGB color space to extract two kinds of features. During the image searching phase, the algorithm will use *OpenCV* brute force matcher to match the features extracted from the images according to the L1 distance. After that, the matched features pairs will be checked by the perspective filter, which checks whether there is projective transformation between the pair of features. Finally, the SIFT match score is calculated as the number of feature pairs after filtering between the query image and the database image. However, if we only use SIFT match score to search the image, the performance is not ideal. Because it is possible that the true positive and false positive images have the same number of matched features. In this case, the algorithm will rank them according to the order in the file loading phase. Therefore, I combine SIFT match score with histogram match score to improve the performance. The color histogram match scores are calculated as:

$$Intersection\left(h(\boldsymbol{q}), h(\boldsymbol{f})\right) = \sum_{i=1}^{bin\_num} \min\{h(\boldsymbol{q})_i, h(\boldsymbol{f})_i\}$$

$$match(\boldsymbol{q}, \boldsymbol{f}) = \frac{Intersection\left(h(\boldsymbol{q}), h(\boldsymbol{f})\right)}{\sum_{i=1}^{bin\_num} \mathrm{h}(\mathbf{q})_{\mathrm{i}}},$$

where $h()$ represents the histogram extractor. The database images are sorted according to SIFT match score in the first priority and color histogram match score in the second priority with descending order.

3. Result Analysis

3.1 Method 1: CNN-based Method

During the experiment, I mainly tested the pretrained-network and the distance function of the algorithm. For the network, I tried AlexNet, VGG, ResNet, and DenseNet. The AlexNet has the best performance. For the distance function, I tried Euclidean distance, cosine distance, and Euclidean times cosine distance. The cosine distance has the best performance. The MAP of this method is shown below:

Average Precision of Q1: 0.8161

Average Precision of Q2: 0.6359

Average Precision of Q3: 0.2268

Average Precision of Q4: 0.6254

Average Precision of Q5: 0.8730

Average Precision of Q6: 0.3584

Average Precision of Q7: 0.2679

Average Precision of Q8: 0.1220

Average Precision of Q9: 0.7741

Average Precision of Q10: 0.6607

Mean Average Precision: **0.536044**

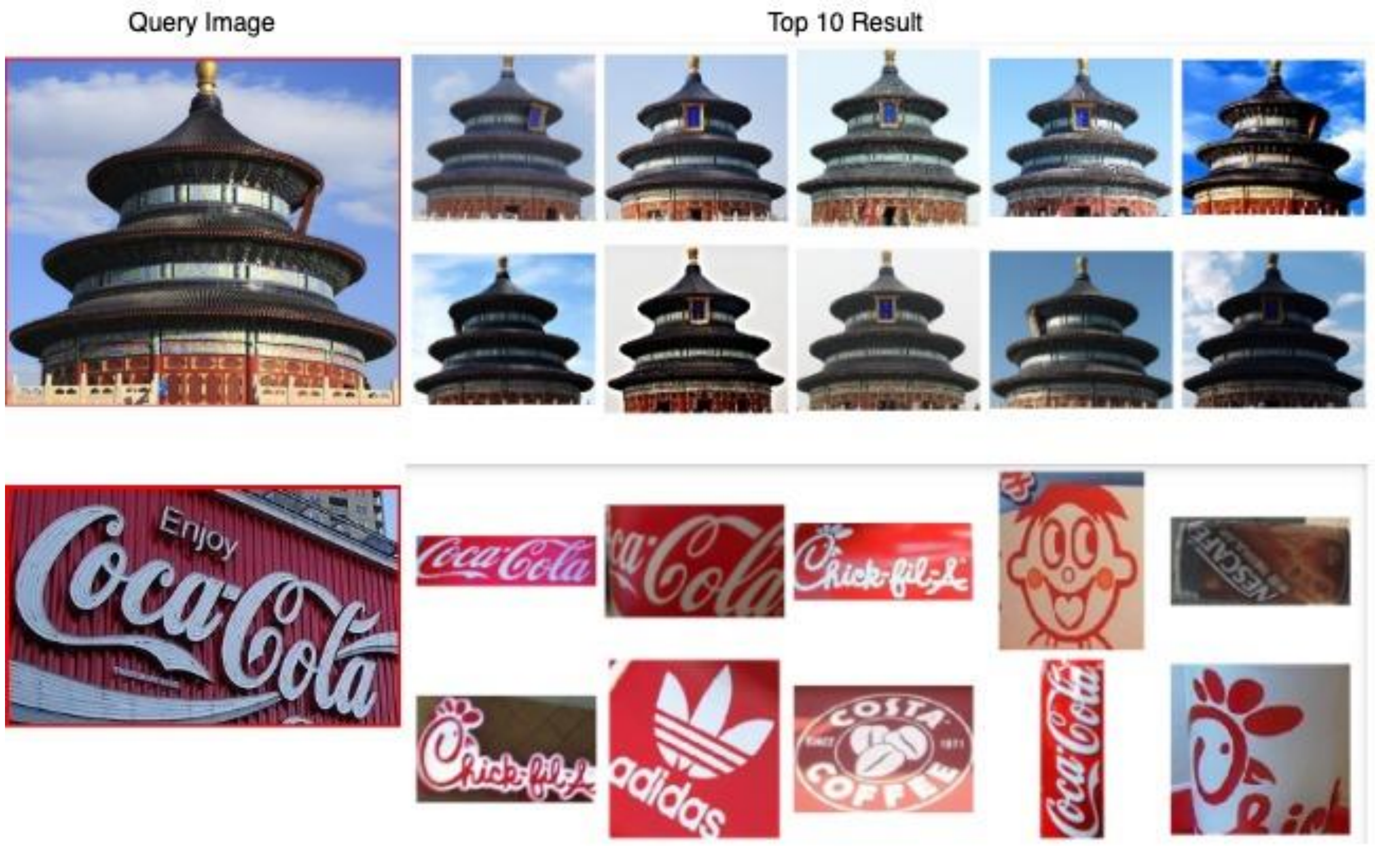

Figure 3: CNN Query Result Examples

The high precision query Q5 and poor precision query Q8 are shown in Figure 3. According to my observation, the precision of CNN is high if the query object is unique enough in its category. For example, Q1, Q5, and Q9 have unique architectural structure although all of them belong to the class "building". However, the Coca-Cola brand is not unique enough for the CNN to recognize in Q8. This might be caused by the pre-trained method. The pre-trained AlexNet is trained for classification problem, so the network tends to classify the objects instead of extracting the significant features, especially for the non-unique objects.

3.2 Method 2: Combined Method

During the experiment, I mainly tested the result of pure SIFT, pure color histogram, SIFT with perspective filter, and SIFT with perspective filter guided by the color histogram. The combined method has the best MAP performance (0.42) compared with pure SIFT (0.25), pure color histogram (0.004), and SIFT with perspective filter (0.4).

Average Precision of Q1: 0.2263

Average Precision of Q2: 0.9517

Average Precision of Q3: 1.0000

Average Precision of Q4: 0.0250

Average Precision of Q5: 0.4187

Average Precision of Q6: 0.9356

Average Precision of Q7: 0.0036

Average Precision of Q8: 0.4081

Average Precision of Q9: 0.0163

Average Precision of Q10: 0.2380

Mean Average Precision: **0.422327**

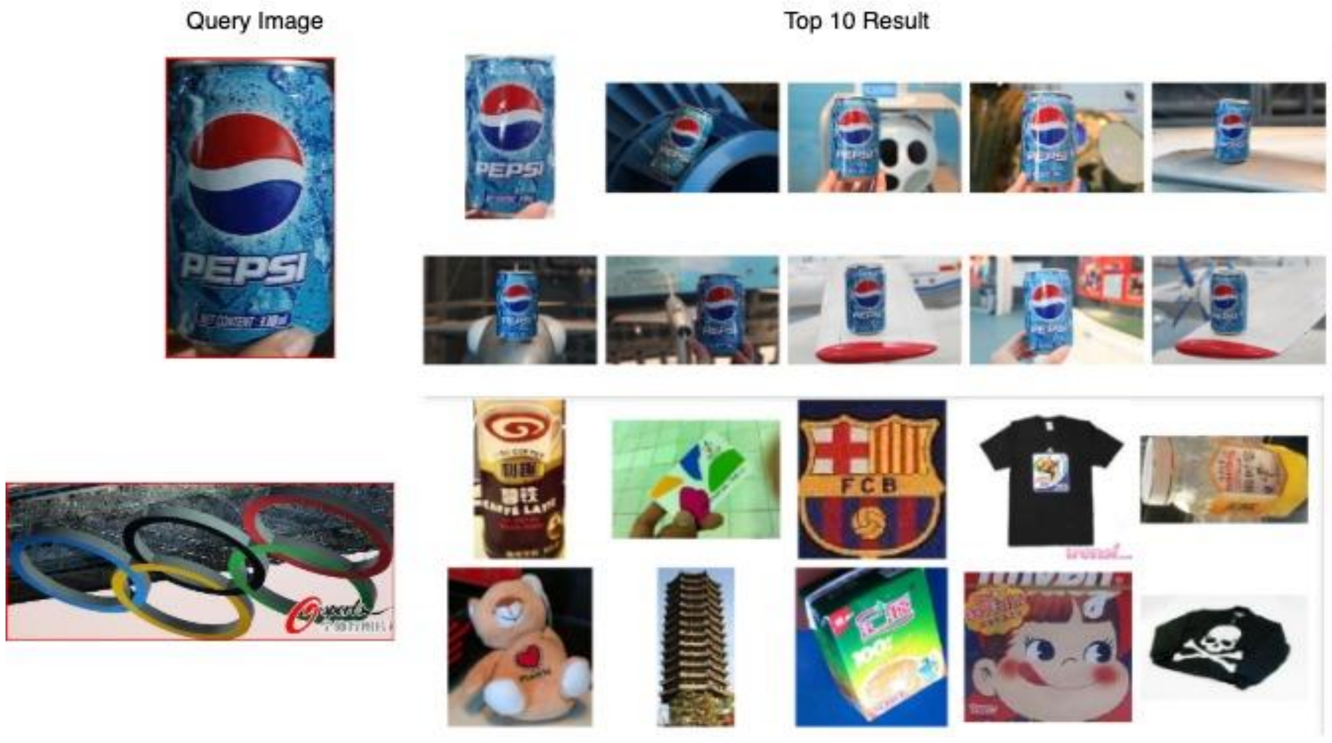

Figure 4: Combined Method Result Examples

The high precision query Q3 and the poor precision query Q7 are shown in Figure 4. According to my observation, the algorithm has a good performance for those images of small items in different environments. Since the object is small, it is relatively easy to classify the foreground and background in the image. For larger objects in different perspectives, the algorithm will be distracted by other objects in the image or watermarks and detect irrelevant features. The color distribution also plays an important role in the high precision, since the tin in Q3 has a more unique color distribution.

4. Conclusion

As a result, we can find that the first algorithm has a higher precision, but the performance is not good enough if the object is not unique enough. However, the second method has a better result for the small, non-unique objects. There are several methods to further improve the accuracy. Firstly, the CNN method is pre-trained for classification problem. If we can change the training method, the accuracy might be significantly improved. Indeed, research works such as [3] modifies the training method and the network structure to improve the MAP to 0.9+. But I think it is a boring act to apply someone else's research to improve the benchmark. The more exciting discovery in this project is the second combined method. **Although the overall accuracy is lower than the CNN method, it provides an elegant and easy-to-implement framework to assist SIFT by the color histogram.** Similar approach can be applied to the CNN method. Different from the SIFT match score (the number of matched features), the CNN distance is continuous. Therefore, it is difficult to combine a second priority algorithm to sort the database images. However, we can choose a hyperparameter $delta$, which represents the if the CNN distances between two database images and the query images are in the same $delta$ range, we can treat the distance as the same during the sorting phase. In this way, the continuous distance space can be transformed to a discrete space.

References

[1] M. Eastwood, "Creating a Visual Search Engine Using PyTorch," 2 July 2020. [Online]. Available: http://www.numerately.com/blog/creating-a-visual-search-engine-using-pytorch/. [Accessed March 2021].

[2] Intel Corporation, Willow Garage, Itseez, "OpenCV-Python Tutorials," Intel Corporation, June 2000. [Online]. Available: https://docs.opencv.org/master/d6/d00/tutorial_py_root.html. [Accessed March 2021].

[3] Filip Radenovic, Giorgos Tolias, and Ondrej Chum, "Fine-tuning CNN Image Retrieval with No Human Annotation," in *TPAMI* , 2018.