Reverse Visual Search based on ResNet50 and FaceNet

Github URL : https://github.com/ls5399/AI_project_2022

Member:

Liangzhou Su : ls5399@nyu.edu
Ziwei Qiu : zq2047@nyu.edu
Zhendong Tan : zt2099@nyu.edu

Introduction

In many domains we are interested in analyzing a “target image”, and then search our database for similar images. For example, this could be a database of images from previous inspections in a manufacturing plant. We might then be interested in finding images of similar objects, or all the historical inspection images of a specific piece of object. With lack of any relevant metadata about the image content, we used computer vision and deep learning techniques to extract useful information without having to manually scroll through all your images.

In this project we built a reverse visual search application with ResNet50 on LFW Dataset and used the performance as the baseline and made improvements by using FaceNet to get better results. What we can then do, is to first convert our target image into a feature vector representation by processing it through our “feature extractor model”. Having done this, we can then perform a similarity search through our database of feature vectors. In the end, we tested the system on images of faces and videos of people to retrieve the 20 most matching images or videos.

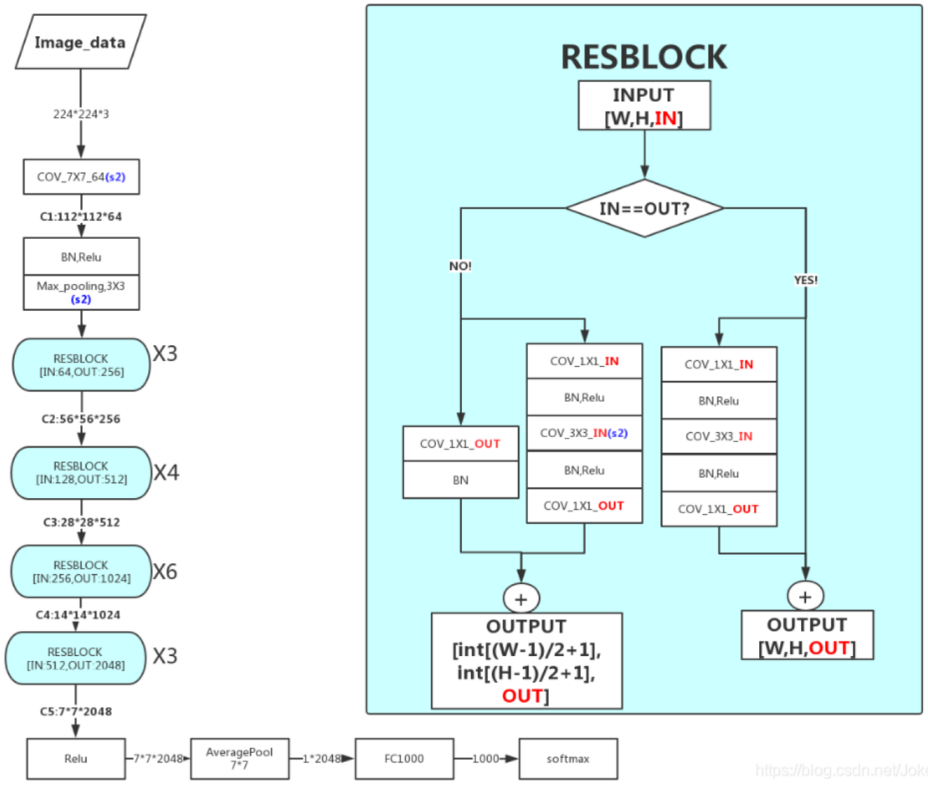
Data

Labeled Faces in the Wild (LFW), a database of face photographs designed for studying the problem of unconstrained face recognition. The data set contains more than 13,000 images of faces collected from the web. Each face has been labeled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the data set. The only constraint on these faces is that they were detected by the Viola-Jones face detector. More details can be found in the technical report below.

Two editions, “All images as gzipped tar file” and “All images aligned with deep funneling”, are used in this project.

Methods and Methodologies

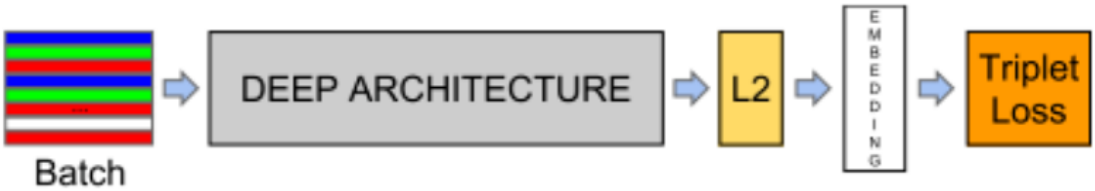
ResNet50:



The core idea of ResNet is introducing a so-called “identity shortcut connection” that skips one or more layers. ResNet50 is a variant of the ResNet model which has 48 Convolution layers

along with 1 MaxPool and 1 Average Pool layer. It has 3.8 x 10^9 Floating points operations. It is a widely used model in face recognition, and thus used as the baseline of our reverse visual search application.

FaceNet:



FaceNet provides an unified embedding for face recognition, verification and clustering tasks. It maps each face image into a euclidean space such that the distances in that space correspond to face similarity. The core differentiator of FaceNet from other models is that it learns the mapping from the images and creates embeddings rather than using any bottleneck layer for recognition or verification tasks.

Similarity search:

Similarity search is the most general term used for a range of mechanisms which share the principle of searching (typically, very large) spaces of objects where the only available comparator is the similarity between any pair of objects. This is becoming increasingly important in an age of large information repositories where the objects contained do not possess any natural order, for example large collections of images, sounds and other sophisticated digital objects.

Experiment

PART I

Baseline Using ResNet50

In the baseline solution, we use ResNet50 as the pretrained model to extract the characteristics of images. In Particular, we are using every layer of the model except the prediction layer to extract the feature vectors of images.

After extracting the feature vectors, we apply Principal Component Analysis(PCA) to the feature vectors to reduce the number of features to 900. This parameter is subject to change. The lower this parameter, the less features we are comparing and may result in worse performance in comparing similarity. If this parameter is too high, overfitting will occur.

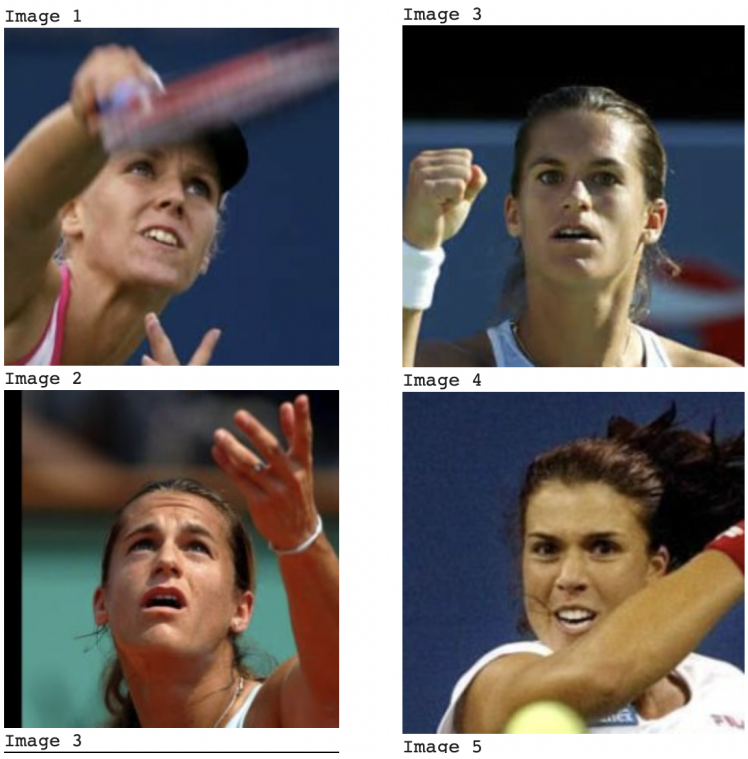
We then get a list of features which contains the vector features of the images. Just like K Nearest Neighbor Algorithm compares the euclidean distances among the clusters, we compare the relative “distance” or “similarity” between the feature vectors (we use **cosine similarity** in this project), and sort the list feature vectors from most similar to least similar.

According to the requirement of the project, we output the most similar 20 images corresponding to the randomly selected query image. A typical image query looks like below. The implementation of our project is all based on **Colab**.

```
q_idx,thumbs,idx_closest=ret(image_paths)
print("Query Image 2")
display(Image(filename=image_paths[q_idx]))
print("\n")
for i,idx in enumerate(idx_closest):
    print("Image %d "%(i+1))
    display(Image(filename=image_paths[idx]))
```

```
Query Image 2
```

After that, we use the idx_closest vector to output 20 most similar faces with the query. The first image is itself, as expected.



We also concatenate 20 most similar images in a single image in AI_final.ipynb for view. But for better visualization purposes, queries and their 20 most similar images are also displayed in AI_final.ipynb.

Similarities(in cosine distance) using ResNet50

Examples:

Image 1

Similarity: 1.0

Image 2

Similarity: 0.5515346527099609

Image 3

Similarity: 0.5201093554496765

Image 4

Similarity: 0.48908478021621704

(for details of 10 random queries, please refer to ResNet50_baseline.ipynb)

From the above pictures we can see that, although the most similar image is the query image itself, the following images have relatively low similarities, and the images are not quite similar to the query image.

Limitations

After seeing some output images by the baseline approach, we can see that the above approach fairly does its job of selecting similar images given the query image. However, it has several limitations.

We can observe that the above algorithm tends to produce similar images, rather than similar faces. Faces in specific backgrounds, faces with exaggerated emotions and faces with certain objects are easily detected by feature extractor. The feature extractor tends to focus on those characteristics in images rather than the characteristics of the faces. This is unwanted because we want to search for people with similar appearances, rather than images with features that are unimportant to us.

The pretrained model of this ResNet50 is designed for classifying images, not people's faces in particular. In order to achieve better performance on the facial field, we are going to deploy a more reliable model which focuses on recognizing faces, such as FaceNet.

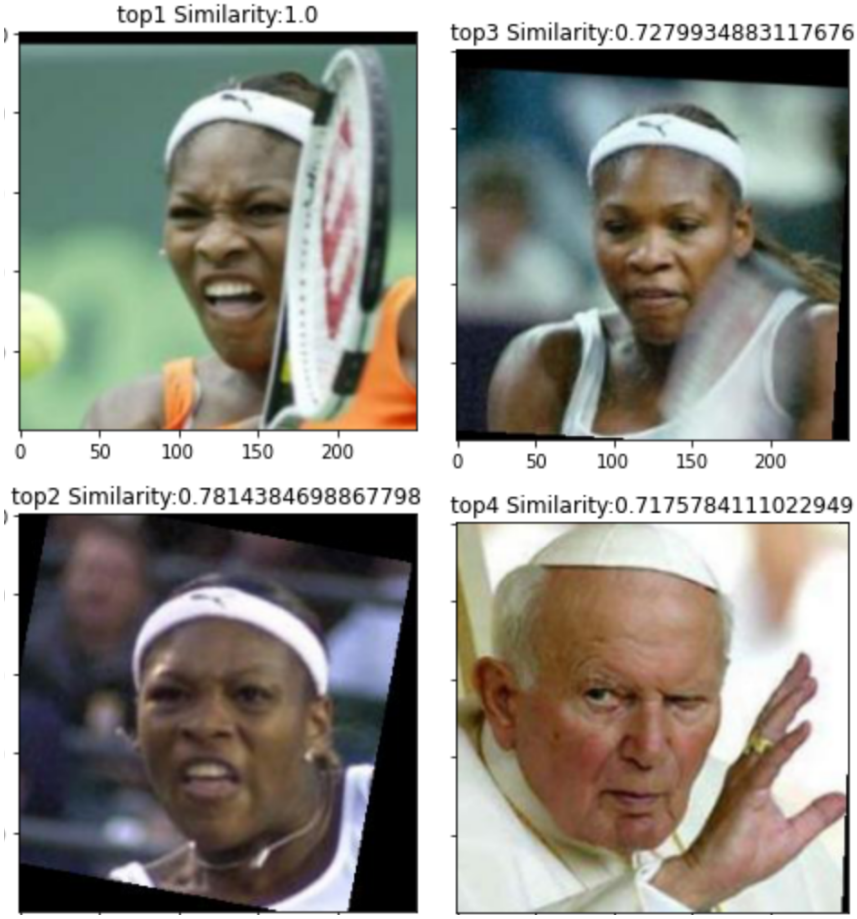
PART II

Improvements from baseline Using FaceNet

FaceNet is pre-trained by a large number of facial images, which makes it obtain a great ability in the field of Facial-related computer vision tasks. In this part, FaceNet will be used as a features extractor. In order to change it into a features extractor, we removed the classification layer of FaceNet.

PCA operation with the same parameters is also used in this part. Furthermore, in order to extract the facial features more efficiently, the facial images in the dataset are aligned. Other operations are similar to the previous part, so we do not introduce them again.

Improvement



(for details of 10 random queries, please refer to Facenet_vision_search.ipynb)

We could find that FaceNet has better performance than Resnet50. FaceNet could capture the facial features more accurately than our baseline model. Also, The corresponding Cosine Similarity is higher by about 10% than the outcome of Resnet50 on average. More example results will be shown in the Colab Output.

All code and results will shown in the GitHub repo: https://github.com/ls5399/AI_project_2022

Conclusion

In this project, we can draw some conclusions:

- The pretrain data will help the model to get a better ability at the beginning. However, we should take care that we should choose corresponding pre-trained models to deal with different types of tasks.
- PCA will help to deal with the overfitting problem. But the super-parameters will affect the performance a lot. We should take care and figure out suitable super-parameters.
- FaceNet will be a better model than Resnet50 in facial-relative computer vision tasks. It has a great performance in the Reverse Image Search problem.

Reference:

[1] Building a visual search application with Amazon SageMaker and Amazon ES | AWS Machine Learning Blog

[2] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[3] Schroff F, Kalenichenko D, Philbin J. Facenet: A unified embedding for face recognition and clustering[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 815-823.

[4] LFW Dataset

[5] The definition of Similarity Search From Wikipedia