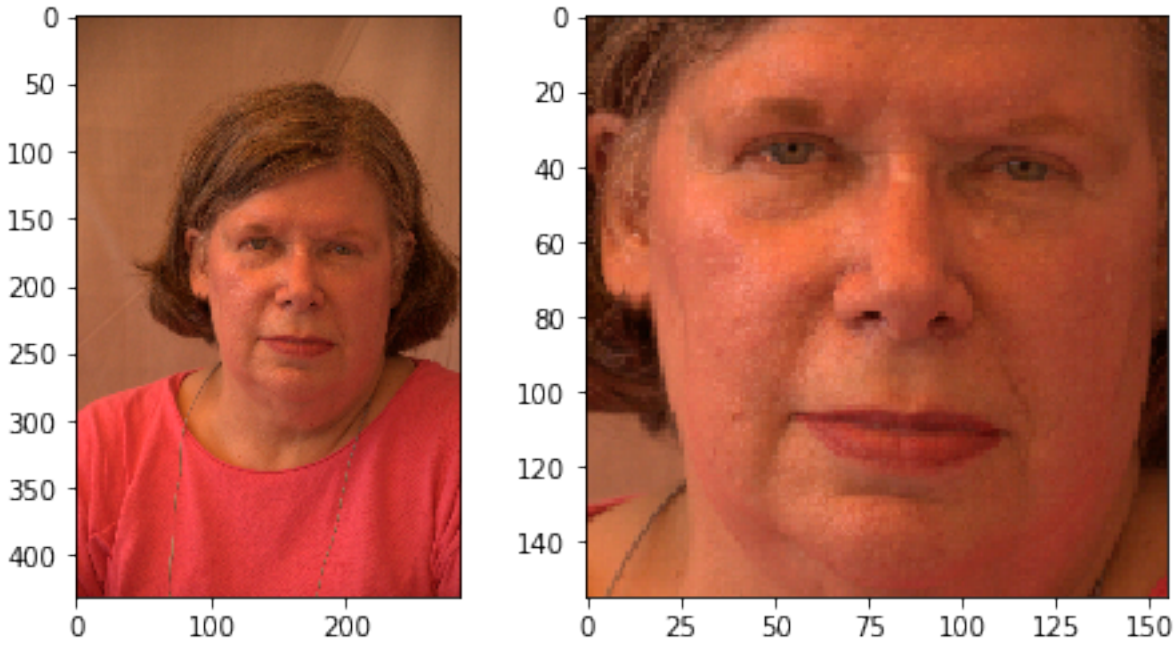
Biometrics HW3

Weixin Jiang
NetID: wjf3669

1. Description about the face recognition system

In this case, we are trying to design a face recognition system and the training and testing dataset are coming from https://drive.google.com/open?id=0B7H6JOp8FYVsb09IVnNMeEd3N00.
This dataset includes faces from 142 different people and each person has 20-200 faces, taken under different conditions, such as facing the camera from different angles, different background, different illumination condition, with or without glasses and so on.
In particular, since some images are in really bad condition, I omitted some bad images and only use part of the whole dataset, which means all training and testing are based on those good face images. Also, I divided the dataset into two parts, one is for training, including almost 80% of images and the other is for testing, including 20% of images.
The procedure of our face recognition system is quite straight forward, first step is we do face detection with some online open source code, in particular, we were using code from https://github.com/ageitgey/face_recognition, which is kind of based on the dlib library. It is one of state-of-the-art face detection and recognition tools, in this case, we just used this powerful tool for face detection.

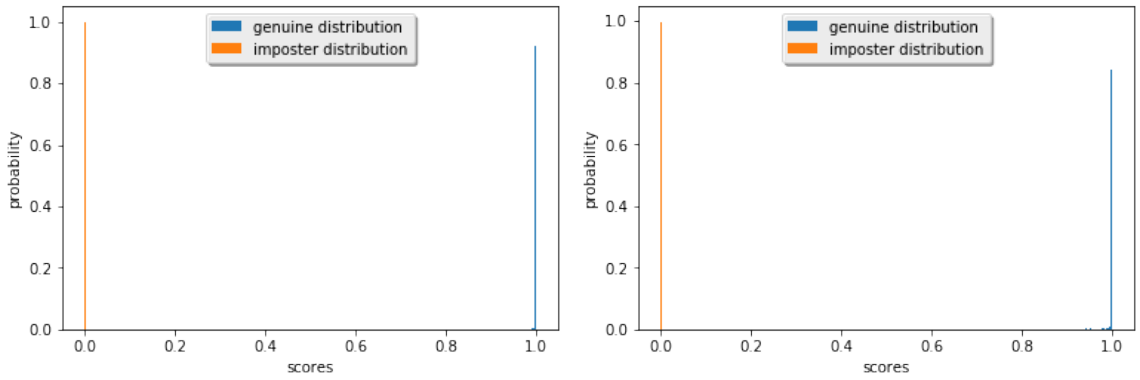


One example is shown above, the left panel is a downsampled face image (because the resolution of the original image is really high!!! So we downsample the image by 10x) and the right panel is the cropped image with the online face_recognition tool. Also this tool provides us a way to screen the dataset, which means we don't use the images whose face cannot be detected by this tool.

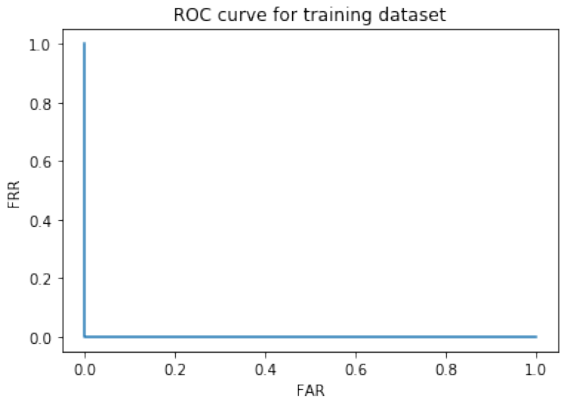
The next step is face recognition part, in practice, what I want to mention here is that the accuracy of face recognition system highly relies heavily on the performance of face detection because we only use the detected faces for training and testing part, which means if the face detection is totally incorrect, then the face recognition accuracy will become terrible. Luckily, the online face recognition tool works pretty good.
In particular, our face recognition system is based on a convolutional neural network, since it is a really power way to extract features and do the classification task. With CNN, the face recognition is really simple, just end to end training. In the next section, we will talk more about some details about our implementation.

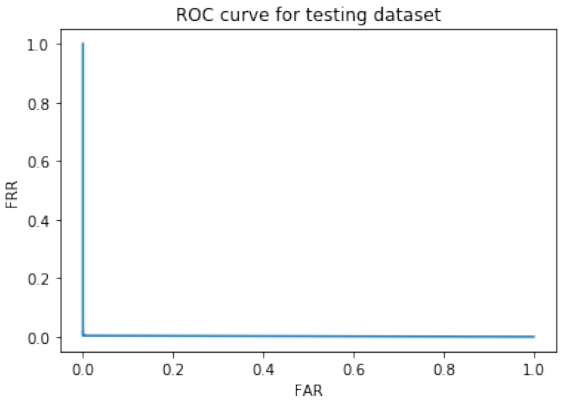
2. Experimental results

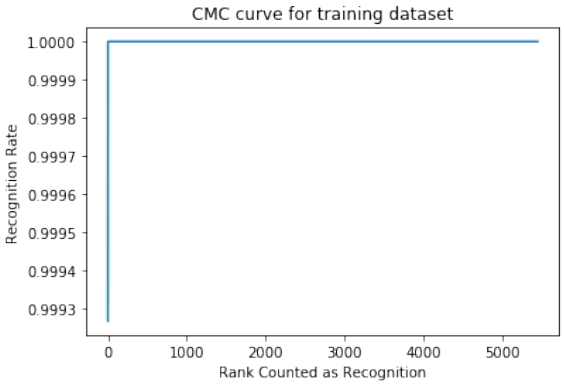
As I mentioned before, our system consists of two parts, face detection and face recognition. For the face detection, there is not much to talk about, first we downsample the original image into a low-resolution image and use the tool to detect the face, if the tool detects a face, then we crop the face image and resize the face image into the resolution of 128x128; if the tool failed to detect a face from the image, then we discarded the image. Also, we randomly select 80% of images as the training dataset and 20% for testing, in the end, we have ~5450 images for training and ~1360 images for testing.
Then for the next part of our system, in particular, we used a 18-layer Residual Network, and for the training, we used the ADAM as our optimizer and chose a learning rate of 0.001, the batch size is 32 and we trained for 100 epochs, details about the architecture can be found in the code. The criterion for our network training is that we first did a log softmax to our output and then used the Poisson negative log likelihood loss as the training loss. For improving the training efficiency, we trained our model a Titan X GPU. Also, in order to avoid the over-fitting issue, we did some data argumentation step, which is that we randomly resized the image into (1~1.35x) and randomly cropped a 128x128 patch from the resized image. We did this data argumentation step for every epoch. As a result, the final loss was reduced to 0.001287 and our prediction of training dataset reaches an accuracy of 99.9266% and our prediction of testing (unseen) dataset reaches an accuracy of 98.8227%.

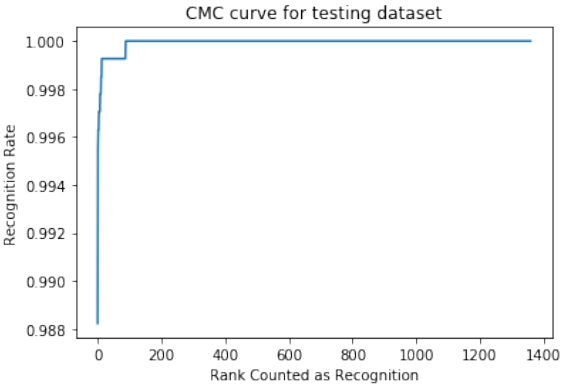


As shown above, the left panel shows the genuine and imposter distribution of our training dataset and the right panel shows the genuine and imposter distribution of our testing dataset. Obviously, our system works pretty good on both datasets.

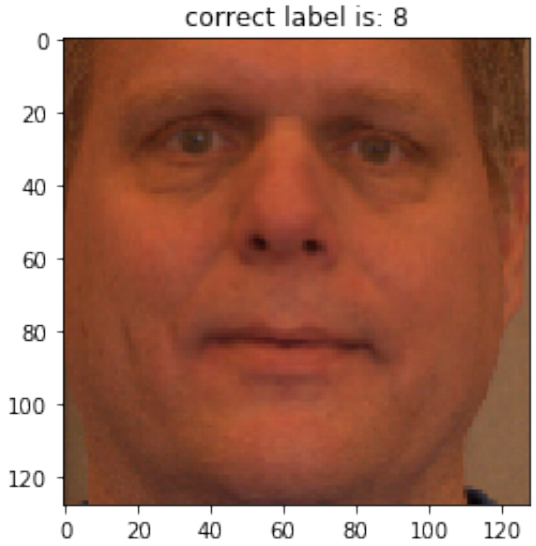


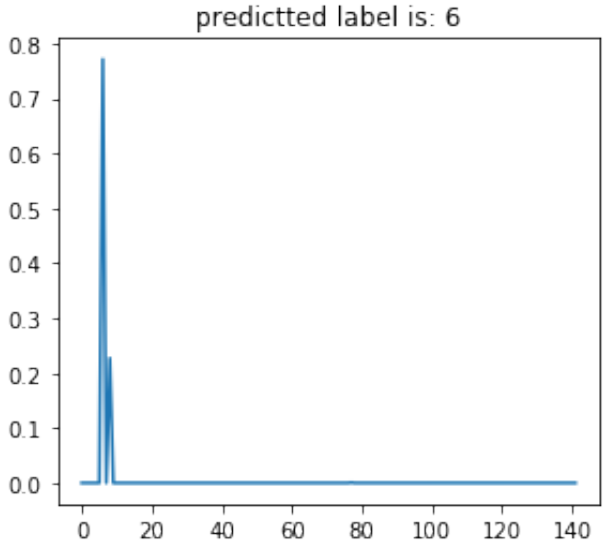


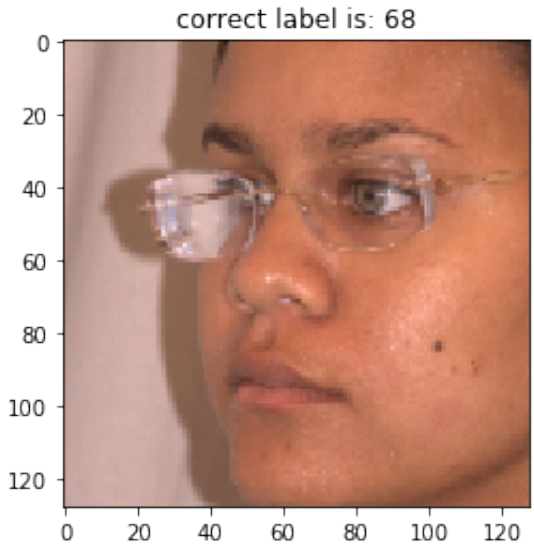


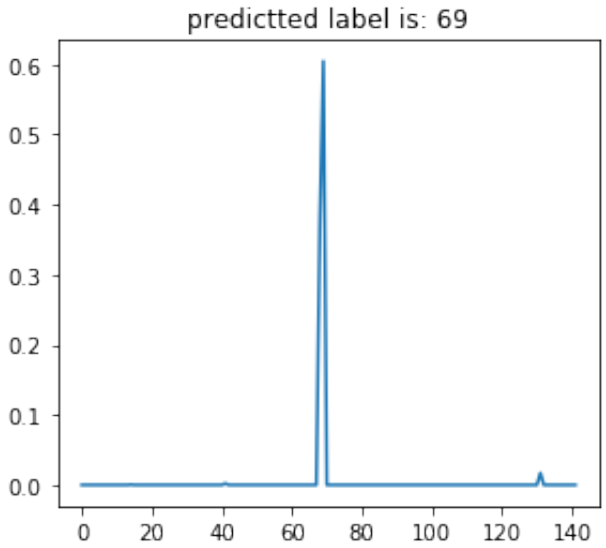


We also plot the ROC curve and CMC curve for both training and testing datasets.
Also, we looked into the incorrect classified images.

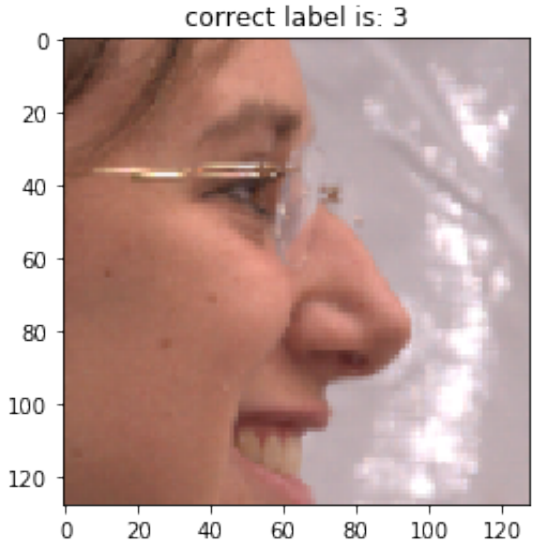


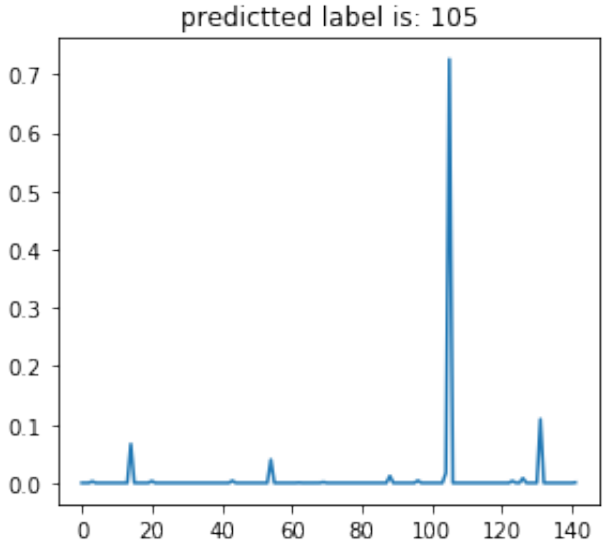


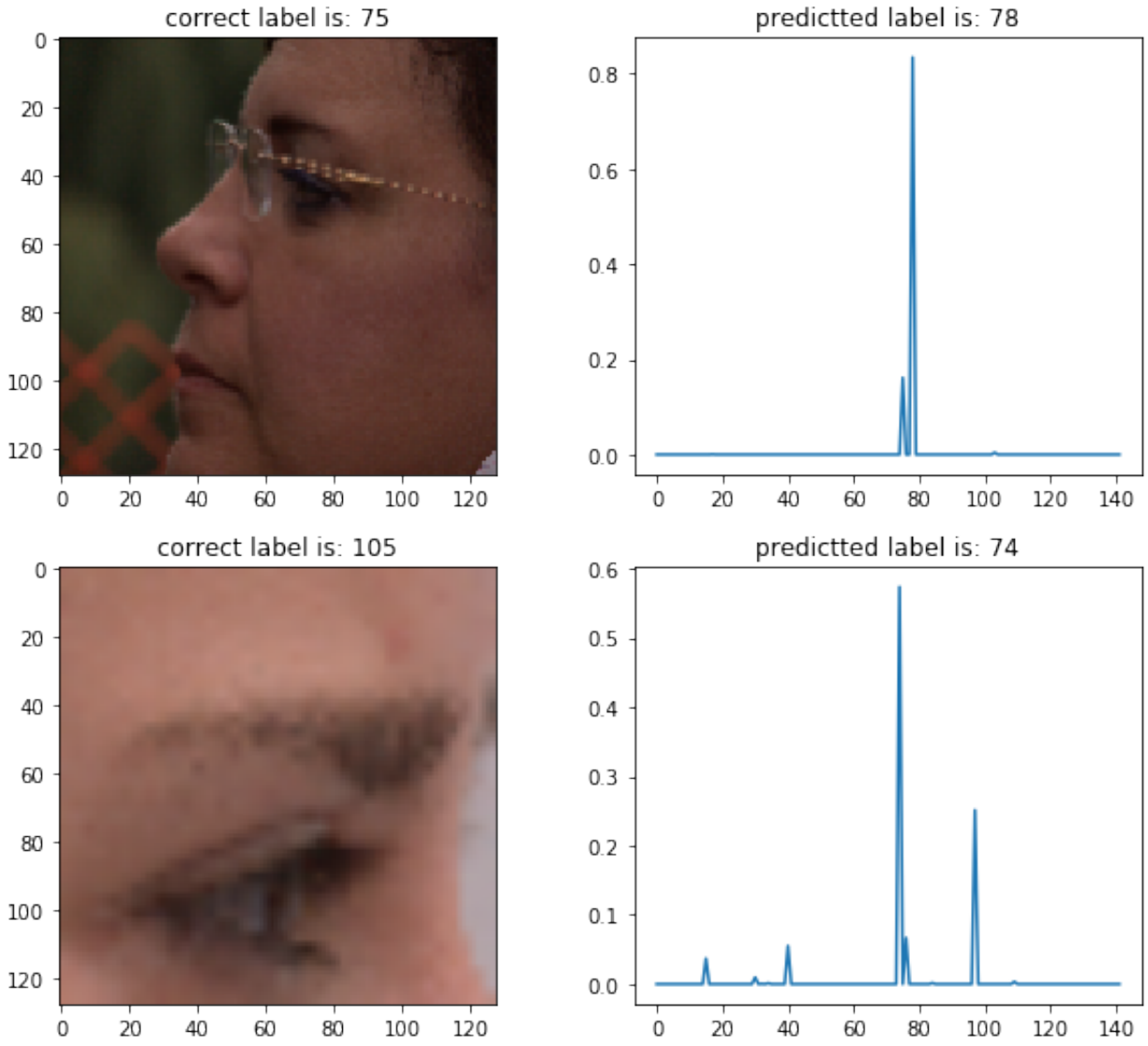




In total, our system incorrectly classified 4 images among 5453 faces in the training dataset, above shows two examples, we may come up with some idea that low-contrast image and shadow may confused our system.







In total, we made 16 mistakes among 1359 images in the testing dataset, above shows some examples, which shows that profile face or low-contrast face is not easy to classify and also the face detection tool does not always work perfect and it leads to misclassification.

3. README for the codes

In order to run the code properly, you may need to install the dlib and face_recognition library, the instruction of installing these two libraries are listed below:
https://github.com/WeixinGithubJiang/face_recognition
Also, you may need to install the Pytorch library to run our code, since our codes are built with the Pytorch library.
If everything mentioned above has been installed properly, then you may run the prepare_dataset.py file, which will generate the training and testing dataset for network training, as a result, it will generate a dataset.h5 file in the same directory as the .py file.
Then if you want to retrain the model, then run the train.py file.

Otherwise, you may run our pre-trained model by running the test.ipynb file, the pre-trained model is stored in the saved_model folder. Since the model is large, because it includes more than 10 million parameters, thus we put the pre-trained model on my Github Repo: https://github.com/WeixinGithubJiang/face_recogntion_project/tree/master/saved_model