Face detection and recognition

Team member:

Name: Yue Lyu Email: yuelyu@ucdavis.edu Name: Yujia Lu Email: yujlu@ucdavis.edu

Introduction

Face recognition is a visual pattern recognition problem rising nowadays. To be more specific, a face recognition system will give the identification result of specific person with input as an image containing this person's face. It has the following four modules: face alignment, face detection, face embedding, face matching(supervised) / face clustering (unsupervised). After fitted different models with best tuned parameters, though all test accuracy for each model are not very high, the SVM seems to perform the best with highest overall accuracy of 95.16% on the whole dataset.

Methodology

Dataset

Labeled Faces in the Wild (***LFW***) is a database of face photographs designed for studying the problem of unconstrained face recognition[1]. 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 people pictured have two or more distinct photos in the data set.

Algorithms/Models

In this part, we only interpret three algorithms that didn't be covered in the lectures.

1). Faces need to be aligned and resize before processing into face detection and embedding. To do face alignment, we used a pre trained model to localize face landmarks which represent salient regions of the face. The model we used combined classic Histogram of Oriented Gradients (HOG) feature combined with a linear classifier. HOG is a feature descriptor for the purpose of object detection. The essential theory behind HOG is to described distribution of intensity gradients or edge directions and get histogram of oriented gradients descriptor which is local object appearance and shape within an image. Thus, using this model, we are able to extract the face landmarks for further calculation[2].

2). Face detection segments the face areas from the background. A pre-trained caffe model, named 'res10_300x300_ssd_iter_140000.caffemodel' is used to perform face

detection on our dataset, which was trained using 'Shot Multi Box Detector(***SSD***)'[3] method. ***SSD*** is designed to perform faster detection without requiring high resolution images and improve accuracy. It is actually a network that base on ***VGG16***[4] to extract feature maps and then added Convolutional neural network after ***VGG***. ***VGG*** was designed for object detection originally and used to train face detection model on ImageNet dataset, which is a dataset of over 15 million labeled images belonging to more than 20000 categories. The key features of the algorithm are that ***SSD*** applies multi-scale feature maps and convolutional detector for detection. It also associates a set of default boxes which are used to be evaluated the offsets of predicted boxes and relative box shape for predicting scores per class that indicate the presence of a class instance in each box. Output of this model returns multiple predictions containing bounding boxes and scores for the presence of the object instances in those boxes.

3). Face embedding is performed though deep Convolutional Neural Network(***CNN***). It embeds an image ***x*** into a d-dimensional Euclidean space represented by $\boldsymbol{f}(\boldsymbol{x}) \in \boldsymbol{\mathcal{R}^d}$. Triplet loss function was introduced in the paper to trains ***CNN*** output to be a compact 128-Dembedding[5]. The network was based on based on GoogLeNet style Inception models(NN4) with 96x96 input size and 22 layers ***CNN*** followed by ***L2*** normalization, which results in the face embedding. This is followed by the triplet loss during training. To define triplet loss, we first have an image x_i^a (anchor)of a specific person is closer to all other images x_i^p (positive) of the same person than it is to any image x_i^n (negative) of any other person. The Triplet Loss minimizes the distance between an anchor and a positive and maximizes the distance between the anchor and a negative.

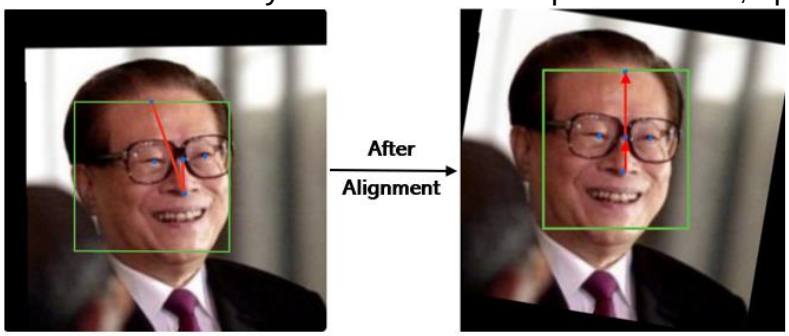
Implementation

Pre-processing

To identify people and match them with labels in the dataset, enough data for each person is needed. Since many people in the data set only have five or less pictures, we only collected images of people who had 20 or more pictures for this project. After filtering the number of images of each person, there are 62 people left with 3023 images. We randomly chose 80% of all images as training set and the remaining 20% as testing set.

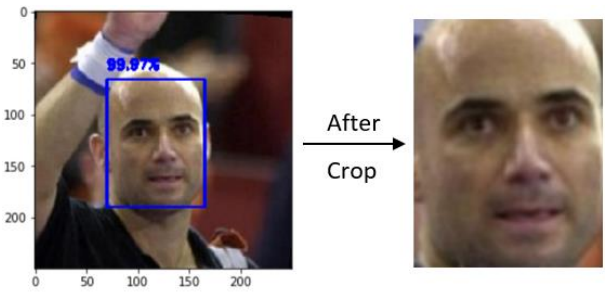
Face alignment

Face alignment aims at achieving more accurate localization faces. We have majorly used ***dlib*** for face detection and facial landmark detection[6], which has no requirement for input image size. The model will return four coordinates locating the green box and another five coordinates representing nose and corner of two eyes. The process of face alignment was shown in the below example. After averaging the corners of each eye, five blue points are highlighted by coordinates of two eyes with the middle point of them, tip of nose and middle point of the green box's top line. Two red lines in the left image are connections of nose and two middle points. The image will then be rotated by a specific angle with "nose" as origin point, which is the inclined angle of two red lines calculating by Law of Cosine. All images were first rotated clockwise. If two red lines overlapped, the product of red lines' length will be equal to the inner product of two vectors originated with "nose" ended two middle points. Otherwise, the image should be rotated counterclockwise.



Face Detection

After face alignment we resized and centered each image before face detection because the input of ***SSD*** model requires a sepcific image size(300x300). As described in the methdology part, the output of ***SSD*** model containing bounding boxes(four coordinates to locate the box) and scores for the presence of the object instances in those boxes(i.e. the probability that detected object is a face). We croppe the part inside the box and then get tightly cropped image of face area(an example shown in left) to feed into the model for feature extraction. Since each image only has one label(name), we assume that there is one and only one face in each image. So, for one image, we only selected the box with the highest score(highest probability to be face).The last step we did here is to resize and



center each cropped face image again to meet the input size requirement(96x96) of face embedding using ***OpenCV***.

Face Embedding

Face embedding will then be performed though deep Convolutional Neural Network with a 96x96 face image input. A 128-dimension vector will be generated to capture the face features by using a pre-trained ***torch*** model called 'openface_nn4.small2.v1.t7' in OpenCV based on the deep ***CNN*** described in the methodology part.

Face Matching

Supervised learning algorithms including SVM, KNN, Random Forest and Neural Network were conducted to train a face identification system. By tuning hyperparameters in each models, we will look at both average cross validation score and accuracy of test set to select candidate models. To train Neural Network, we randomly chose activation function and tried different number of hidden layers, batch size and learning rate.

Face Clustering

We also tried to face clustering to capture differents face expressions by hierarchical clustering and k-means. The input was 128D embedding and since the embedding lends itself to be used with the same identity, we only chose one image of each person. Here we just tried to cluster all images into four classes(each row for one class). The result turned to be interesting. The first row shows a "scornful" expression and "happy" face in second row, third row shows "angry" faces and "funny" face in fourth row.

Result

Performance and evaluation

We tuned hyperparameters for each model, selected final models using 5-fold crossing validation with accuracy score. The first three accuracy scores are the best accuracy for each model during tuning, and after we fit these best models, we fit the whole dataset again and examined the accuracy. It's clear that SVM is the best model with both high train accuracy and test accuracy. To address the problem of unbalanced number of images for each person, we tried to separate the dataset into 3 groups, ie people in large group has more than 100 images, then fit models with these groupped data. The result supports that larger numbers of images increases the overall accuracy of fitting.

	SVM	Random Forest	KNN	DNN
CV Accuracy	0.542	0.431	0.4374	
Train Accuracy	0.967	1.0	0.5939	0.5726
Test Accuracy	0.558	0.444	0.4374	0.5676
Overall Accuracy	0.9516	1.0	1.0	

References:

[1]http://vis-www.cs.umass.edu/lfw/#explore

[2] https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients

[3]Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, SSD: Single Shot MultiBox Detector, arXiv:1409.1556 [cs.CV]

[4]Karen Simonyan, Andrew Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv:1409.1556 [cs.CV]

[5]Florian Schroff, Dmitry Kalenichenko, James Philbin, FaceNet: A Unified Embedding for Face Recognition and Clustering, arXiv:1503.03832 [cs.CV]

[6]https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/