Machine Learning Engineer Nanodegree

Capstone Proposal

Xinkun Chu
March 2, 2017

Domain Background

Face recognition is a hot topic in the field of computer vision. It has wide applications around us. Computers and smart phones recognize users' faces to unlock the devices. Smart cameras use it to focus on the faces in portrait photography. Facebook use the technique to speed up tagging people in photos. “Swiping face” is considered more safe and convenient than “swiping card” where people can pay by simply taking a picture. However, face recognition is far from perfect. Different lighting, makeup, decorates, change in age, change in facial expressions and low resolution image all make the face recognition task difficult. More efforts are needed on the topic to get face recognition technique further applied in everyday life.

The study on face recognition has a history of more than 50 years. The first stage dates from 1960 to 1990. Face recognition was studied as a branch of general pattern recognition, and mainly used geometric feature based method[1]. In the 1990s, face recognition came to the second stage with fruitful results. Researchers came up with new algorithms like Eigenface[2] and Fisherface[3]. Appearance based and statistical pattern recognition based algorithm showed good performance on well-controlled, front- face images taken in the lab. In the current stage, researchers are working to improve the robust and accuracy of face recognition under complex circumstance.

Problem Statement

The problem of face recognition can be generalized as:
Given input face images and a database of face images of known individuals, determine the identity of the person in each input images.

Datasets and Inputs

To begin with, the Olivetti face dataset[4] will be used. The dataset contains ten different images for each of 40 distinct subjects taken at AT&T Laboratories Cambridge. As described on the original website: For some subjects, the images were taken at different times, varying the lighting, facial expressions (open/closed eyes, smiling/not smiling) and facial details (glasses/no glasses). All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement).

Then, the Labeled Faces in the Wild dataset[5] will be used. The data set contains more than 13,000 images of faces collected from the web, with each face labeled with the name of the person. 1680 people pictured have two or more distinct photos in the data set. The funneled version of the dataset with aligned images will be used.

The Olivetti face dataset is taken between April 1992 and April 1994, during the second stage of the development of face recognition (as described in the history part), when face images in research were under well-controlled circumstance. It would be a good start point to train a naive toy model for face recognition and check the performance on this simple dataset. The labeled Faces in the Wild dataset was designed to study unconstrained face recognition, with images taken from internet rather than lab. It's a more challenging task of face recognition under complex realistic circumstance, and will be the focus of this project.

Solution Statement

Convolutional Neural Network model will be used to solve the problem. CNN can automatically learn features to capture complex visual variations in the images. The person identity can be one-hot encoded as the label. The model can be trained with the face images of known individuals with labels, and calculate the probability of the person in test images belonging to each identity. The identity with the highest probability will be taken as the predicted identity for the person in the image.

Benchmark Model

The example model of face recognition using eigenfaces and SVMs from scikit-learn[6] will be used as the benchmark model. The example model uses the labeled faces in the wild. It uses PCA for unsupervised feature selection and dimensionality deduction. Eigenfaces are the components of the PCA. Then it trains SVM using GridSearchCV for optimization.

First, I'll follow the method applying the model to the Olivetti dataset. The performance of the toy CNN model will be compared with this benchmark model.
For the LFW dataset, the example model only contain 7 people with images for each person more than 70. The averaged precision, recall of the model's predication for these people is around 80%. The example model will be extended to include more people with less face images in the dataset. Then, a more complex CNN model with the same data subset as this benchmark model will be trained. The average accuracy, precision and recall for both models will be calculated and compared.

Evaluation Metrics

During the training, the mean-squared-error (MSE) will be used to check the performance of the model. MSE is defined as:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\widehat{Y_i} - Y_i)^2$$

N is the size of the test sample, $\widehat{Y_i}$ is a vector of predictions at the current step, and Y_i is the vector of actual value(The person name are one-hot encoded).

Accuracy, precision and recall will be used as more intuitive metric to judge the performance of the model on test dataset. Accuracy is defined as the ratio of number of right identified person to total number of person to identify in the test dataset. Accuracy shows the overall effectiveness of the model and tells the average probability of correct identifying a person. But accuracy alone can be biased (the accuracy paradox) for the class imbalance in the LFW data, where several people have more face images than others. Precision and recall will also be used. Precision shows the class agreement of the data labels with the positive labels given by the model. Recall shows the effectiveness of the model to identify positive labels. Precision and recall can be expressed as:

$$\text{precision} = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l}(TP_i + FP_i)}$$

$$\text{recall} = \frac{\sum_{i=1}^{l} TP_i}{\sum_{i=1}^{l}(TP_i + FN_i)}$$

where TP_i is the true positive for class i, FP_i is the false positive for class i, FN_i is the false negative for class i, l is the number of class.

Project Design

Libraries and tools:
Python 2.7
Jupyter Notebook: create and share documents that contain live code, equations, visualizations.
Scikit-learn: simple and efficient open-source tools for data mining and data analysis
Tensor-flow: open-source software library for Machine Intelligence.
Keras: high-level neural networks library, written in Python and capable of running on top of either TensorFlow or Theano.

Workflow:
To start with, a naive toy CNN model will be trained and tested on the Olivetti face dataset. Sklearn.datasets has provided fetch_olivetti_faces() to get the dataset. Keras based on Tensorflow will be used to build the model. Because the dataset is taken in the lab with dark homogeneous backgrounds, and subjects in upright, frontal position, it's expected that a simple CNN model can perform well. The results will also be compared to a benchmark model using the SVM.

Next I will deal with the labeled Faces in the Wild dataset. The example model provided by scikit-learn using SVM only contains people with a minimum face images of 70. I'll decrease the minimum face images from 70 to 25. After splitting the dataset to training, validation and test, there will be on average at least 15 images for each person to train the model. The example model will be extended with this larger dataset and used as the benchmark model.

Then, CNN model will be trained with the same subset data as the benchmark model. The dataset will be divided into training set, validation set and test set. The images will be resized to the same size as used in benchmark model. Person's name will be represented by numbers and use one-hoe encoding. More layers compared with the toy CNN model may need to be added. The hyper-parameters of the model, like number of convolution filters, stride size, dropout rate, will be adjusted according to the metric performance on the validation dataset. The optimized parameters will finally be used in training the model. The final metric performance on the test set will be compared with the benchmark model.

The CNN model's performance may be affected by the fact that it may capture irrelevant features in face recognition with the entire face images. Depending on the previous results, I would consider to apply the facial point detection, provided by the paper.[7] The technique can capture the positions of the five facial points in the face images, two for eyes, one for nose and two for mouth. A possible attempt is to captures the eyes area from the raw face images and train the CNN model with this area. It would be interesting to check whether the model trained with the eye area has better performance over the model trained with entire face region.

Reference

[1] W Zhao, R Chellappa, PJ Phillips, A Rosenfeld, Face recognition: A literature survey, ACM computing surveys (CSUR) 35 (4), 399-458 (2003)

[2] M. Turk, A. Pentland , Face recognition using eigenfaces, Proc. IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–591 (1991).

[3] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman, Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. 19, NO. 7, JULY 1997

[4] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

[5] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49, October, 2007

[6] http://scikit-learn.org/stable/modules/generated/sklearn.datasets.fetch_lfw_people.html

[7] Y. Sun, X. Wang, and X. Tang. Deep Convolutional Network Cascade for Facial Point Detection.
In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013