Face Recognition

Yiyi Liu[1] and Meng Pang[2]

[1] University of Waterloo, Waterloo, On, N2L3G1, Canada
Email:y864liu@uwaterloo.ca
Student ID: 20625436
[2] University of Waterloo, Waterloo, On, N2L3G1, Canada
Email:meng.pang@uwaterloo.ca
Student ID:20645779

Abstract. Face recognition remains a challenging problem till today.We have independently implemented two classification systems SVM(support vector machine) in Matlab and CNN(convolutional neutral network) in Python. And we successfully applied these methods to the face recognition problem. To examine the correctness and accuracy of these two algorithms, we test our systems on Olivetti Faces Database. Our experimental results show that our systems can attain above 90% in accuracy rate.

Keywords: Face Recognition; Classification; SVM; CNN

1 Introduction

Face recognition has developed into a challenging research area in pattern recognition and computer vision due to the wide variation of face appearances, illumination effect and the complexity of the image background. In the past year, the performance of face recognition algorithms increased in a large margin. It is increasingly investigated by Google, Facebook, Intel, Accenture, IBM, DeepMind, and many other companies. Recognition under constrained environment is quite satisfactory, like numerous models have been applied to LFW, the hardest face dataset at present. For example, the Support Vector Machine (SVM) is one of most widely used classifiers in face identification. And convolutional neural network (CNN) is the state-of-the-art deep model for face recognition and image analysis.

SVMs deliver state-of-the-art performance in real-world applications such as text categorisation, hand-written character recognition, image classification, biosequences analysis, etc., and are now established as one of the standard tools for machine learning and data mining. Face recognition is a K class problem. where K is the number of known individuals; and support vector machines (SVMs) are a binary classification method. By reformulating the face recognition problem and reinterpreting the output of the SVM classifier. we developed a SVM -based face recognition algorithm.

Neural network is a very powerful and robust classification technique which can be used for predicting not only for the known data, but also for the unknown data. It works well for both linear and non linear separable dataset. NN has been used in many areas such as interpreting visual scenes, speech recognition, face recognition, finger print recognition, iris recognition etc. A typical neural network is using multilayer perceptron (MLP). This method, however, has massive connection. It also ignores the neighbouring pixels and only handles classification.

In this project, SVM and CNN methods of face recognition is used to develop a real time face recognition system. And We have used the ORL database which contains a set of faces taken between April 1992 and April 1994 at the Olivetti Research Laboratory in Cambridge, to test these two algorithms to recognize the faces and do comparisons and conclusions.

2 Literature Review and Related Work

Recently, face recognition has been widely studied, and thus, some machine learning methods has been improved conspicuously. There are different mechanism for detecting face, such as component-based or global approach[3]. For global method, the trainer analyses the whole face image and get all gray scale values as a single vector to be classified by SVM classifier[3]. Compared with global approach, component-based methods allow flexible geometrical relation between the components in the classification stage, which can compensate the changes of pose and angle of face[1]. In these approaches, face recognition was performed by matching templates of principal facial regions separately [7][9].

Convolutional Neural Network (CNN) was first proposed by Yann LeCun. The proposed method has a very wide range of applications; such as face detection and recognition, gender recognition, object recognition, character recognition and texture recognition. However, [4] presents a hybrid neural-network model combining local image sampling, self- organizing map (SOM), and a convolutional neural network. The convolutional neural network is claimed to perform well for providing partial invariance to translation, rotation, scale, and deformation, as well as extracting successively larger features in a hierarchical set of layers. [8] proposes to learn a set of high-level feature representations through deep learning for face verification.

3 Approaches

3.1 SVM Method

3.1.1 Description For the SVM approach, the system reads the whole grey images as samples and get all of grey pixels of rows and columns of images. Since the database ORL face images are all of the same size(112*92), we could

get uniformed size of samples matrix.
For training part, the system saves the pixels of rows and columns of an image as according array, loads all of the training images and gets complete original training samples. After loading original data, reducing dimensionality is essential for training classifier. Since PCA works fast for preprocessing dataset as well as remaining the significant components for samples, we use PCA to make dimension reduction with parameter k as target dimension.
The following step is to classify the processed dataset with SVM approach. SVM classifier has several different kernel functions to perform representation of original dataset. For **linear**, **polynomial** and **RBF**, there are different performance. Compared with **linear** and **polynomial**, **RBF** performs better in flexibility because the parameters are more than linear and polynomial. After getting classifier model, we use it to test remained grey scale images and then get accuracy for the whole test case.

3.1.2 Implementation

This part would introduce the details of implementation of SVM approach for face recognition

Software: Matlab

Data Structure

1. Whole dataset: ORL frontal face images
2. Training dataset: First 5 images for everyone
3. Testing dataset: Last 5 images for everyone

This part uses ORL face images as dataset. This dataset uses 400 frontal images of 40 persons taken at different time. Since these images are uniformed to same size and grey scale, they can be loaded directly into the system and transferred into matrix for processing, training and testing.

Program Structure

SVM approach organizes as follows:

1. Loading training images as matrix: The system uses **imread()** function to load grey scale images to .mat format for next processing.
2. Using PCA to reduce dimensionality: The PCA method is used for reducing dimensionality to target dimensionality. In this project, SVM algorithm would get dimensionality disaster without dimension reduction. We set the target dimensionality as k and reconstruct face images as **eigenfaces** and recognizing different persons images.
3. Training SVM classifier: Using **RBF** as the kernel function to training half face image matrix.

4. Testing remained part: Reducing dimension for test data and making test with built classifier to another half images.
5. Calculating result: Calculating the accuracy

Data Representation

1. **recognize**: matrix for classifier
2. **multiSVMstruct**: classifier model
3. **scaledtestface**: scaled face data after normalized
4. **testface**: original test data matrix
5. **realclass**: the real class label for test data
6. **parameters**: k as target dimension for PCA, γ and C for **RBF** kernel function

Above data representations are critical for training classifier and test dataset normally. The parameter γ and C are for SVM kernel function **RBF**. Thus, there are different classify accuracy for different parameters.
After reducing dimensionality, the machine could figure out the first k principal components and using them to get **eigenface**. The **eigenfaces** can reconstruct training images by multiplying according co-efficiency at some degree.

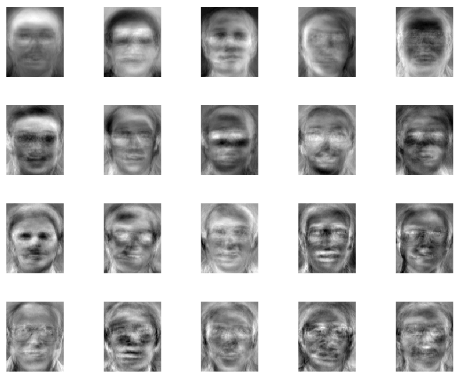

Fig. 1. Eigenfaces

Figure 1 indicate 20 different eigenfaces drawing from preprocessed image dataset when $k = 20$, actually, they can be seen as 20 eigenvectors and used for reconstructing original sample data somehow.
Figure 2 show the reconstructed 1^{st} image for 1^{st} person and the real one. Figure 3 indicates the comparison of the original image and reconstructed image for the

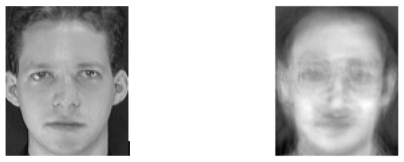

Fig. 2. Original and Reconstructed image for 1st person

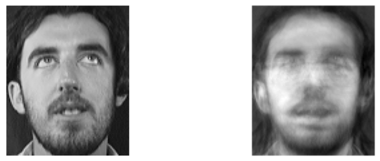

Fig. 3. Original and Reconstructed image for 11th person

11^{th} person.
From the two pairs of images, we can almost tell that the sketch of original images can indicate the main feature of every face at some degree, although they are not accurate. The accuracy for eigenfaces are determined by the reduction dimensionality method and the degree of reduction. For this project, 20 is the target dimension for reducing dimensionality. Obviously, the performance would be better for more remained dimensions, however, higher dimensionality also increases the complexity and decreases the efficiency.

3.1.3 Testing Testing dataset on the SVM classifier model trained previously, we have different parameters values for γ and C, for classifier kernel function. In this project, we focus C at 128 since it is the boxconstraint Last half face images are the testing dataset. The **testface** stores all information of images. The following are the classification results for different parameters γ and C. Table 1 is the results for testing.

From table 1 we can figure out that the classifier performs better when the dimensionality remained more, especially when the dimensionality is 80.

k / γ	0.0010	0.0030	0.0050	0.0080	0.01
20	83%	83.5%	82%	83.5%	83%
30	85.5%	85%	84%	84.5%	84.5%
40	85.5%	86%	86%	85.5%	87.5%
50	84.5%	85%	87%	88%	86.5%
60	85%	87%	88%	87.5%	87.5%
70	86.5%	86.5%	88%	88%	88.5%
80	87%	89%	88%	88%	88%

Table 1. Accuracy for different parameters

3.2 CNN Method

3.2.1 Description **Neural network** is a very common method in pattern recognition field. Neural network is ideal in handling non-linear problems due to its ability to learn. A typical neural network is using multilayer perceptron (MLP). This method, however, has massive connection. It also ignores the neighbouring pixels and only handles classification.In this section, we briefly describe the Convolutional neural networks. The focus is to inspect its internal structure to provide insights into its respective strengths and weaknesses on the present vision task, e.g face recognition problem.

Yann LeCun and Yoshua Bengio introduced the concept of CNNs in 1995. A convolutional neural network is a feed-forward network with the ability of extracting topological properties from the input image. It extracts features from the raw image and then a classifier classifies extracted features. CNNs are invariance to distortions and simple geometric transformations like translation, scaling, rotation and squeezing.

Convolutional Neural Networks combine three architectural ideas to ensure some degree of shift, scale, and distortion invariance: local receptive fields, shared weights, and spatial or temporal sub-sampling [5]. The network is usually trained like a standard neural network by back propagation. CNN layers alternate between convolution layers with feature map $C^i_{k,l}$ shown by

$$C^i_{k,l} = g(I^i_{k,l} \otimes W_{k,l} + B_{k,l}) \tag{1}$$

And non-overlapping sub-sampling layers with feature map $S^i_{k,l}$ shown by

$$S^i_{k,l} = g(I^i_{k,l} \downarrow W_{k,l} + Eb_{k,l}) \tag{2}$$

Where $g(x) = tanh(x)$ is a sigmoidal activation function, B and b the biases, W and w the weights, $I^i_{k,l}$the ith input and $\downarrow$ the down-sampling symbol.$E$is a matrix whose elements are all one and $\otimes$ denotes a 2-dimensional convolution. Note that upper case letters symbolize matrices, while lower case letters present scalars [6].

A convolutional layer is used to extract features from local receptive fields in the preceding layer. In order to extract different types of local features, a convolutional layer is organized in planes of neurons called feature maps which are responsible to detect a specific feature. In a network with a 5×5 convolution kernel each unit has 25 inputs connected to a 5×5 area in the previous layer, which is the local receptive field. A trainable weight is assigned to each connection, but all units of one feature map share the same weights. This feature which allows reducing the number of trainable parameters is called weight sharing technique and is applied in all CNN layers. LeNet5 [5], a fundamental model of CNNs proposed by LeCun, has only 60,000 trainable parameters out of 345,308 connections. A reduction of the resolution of the feature maps is performed through the subsampling layers. In a network with a 2×2 subsampling filter such a layer comprises as many feature map numbers as the previous convolutional layer but with half the number of rows and columns.

In the rest of this section LeNet5 which is a particular convolutional neural network is described. LeNet5 takes a raw image of 32×32 pixels as input. It is composed of seven layers: three convolutional layers (C1, C3 and C5), two subsampling layers (S2 and S4), one fully connected layer (F6) and the output layer. The output layer is an Euclidean RBF layer of 10 units (for the 10 classes) [2]. These layers are connected as shown in Fig 4.

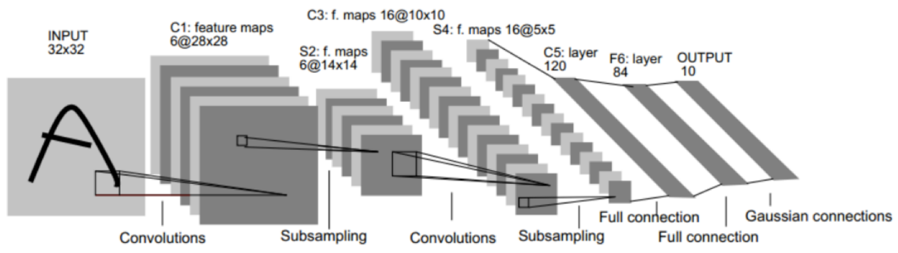

Fig. 4. LeNet5 Architecture

As shown in table 1 the choice is made not to connect every feature map of S2 to every feature map of C3. Each unit of C3 is connected to several receptive fields at identical locations in a subset of feature maps of S2 [2] [5].

3.2.2 Implementation In this part, Convolutional Neural Network (CNN) method is implemented by Python to develop a real time face recognition system. Here CNN is using LeNet-5 architecture. The proposed method is a simplified version of CNN, with lesser layers. It has a complex algorithm but with minimal preprocessing steps.

The objectives of the CNN method part:

- To develop a face recognition system by using CNN.
- To merge the preprocessing stages that has been designed in Pillow with the CNN that has been designed in Python language.
- The system is based on 40 images from ORL database to verify the correctness of my CNN program and examine the accuracy of this method.

Software:

- Python, numpy, Pillow, ePickle, theano

Data Structures:

- Face image consists of 400 images of 40 different people. The total pixels in the whole image is 1190 x 942, and each face image is 57 pixels by 47 pixels.

Data Representation:

- Faciel images are stored at a 8 bit gray scale. Data comes from the Oivetti database at ATT.

Program Structures:

- *Read Olivetti Faces image and generating data convenient for training.* The whole image is read by Pillow library and then transferred into numpy arrays. Each face image is saved into a vector and corresponding labels are appended with it. As a result of this preprocess step, 400 vectors wich represents data in each face images and corresponding label vector are generated.

- *Split dataset into training data, validation data and test data.* The training data consists of 320 face images(vectors) and the validation data consists of 40 face images, as well as the test data. Note that the splitting is not random, for validataion and test data we manually pick 40 images which belong to 40 different faces. While each sample comes from a random selection from the 10 faces belonging to the same person.

- *Build CNN model with the optimization algorithm.* The CNN model consists of 5 layers: one input layer, two LeNet Convolution Pool Layers, one hidden layer and one logistic regression output layer. The algorithms applied in the model is Minibatch Stochastic Gradient Descent(MSGD) algorithm which utilize the gradient descent algorithm to update the parameters in a stochastic fashion.

- *Train and validate the model to apply facial recognition for Olivetti Faces.* In this phase, the program train the CNN with the input of number of epochs, learning rate, pool size, as well as number of kernels, aiming to find the best set of parameters of the convolutional network.

3.2.3 Testing

In the process of training, validating and testing the CNN model, different parameters need to be tuned. These parameters include the learning rate, batch size for stochastic gradient descent algorithm, number of epochs for training as well as numbers of convolution kernel in both LeNet Convolution Pool Layers. number of epochs is set to be big enough to see the optimal error rates, it is set to be 200, but in most cases the optimal error is achieved at 40 epochs. For the pool size for the two convolutional layers, we set it to be 2 x 2 since it is enough for the relative small image of 57 x 47 pixels for a sample. As for batch size, it is set to be 40 since other choices such as 1, 2, 5, 10 and 20 makes the validation error to be 97.5% which is pretty much useless. The following records the the parameter tuning the learning rate and number of convolutional kernels.

– Learning Rate: fixing 20 kernels for first convPool layer and 50 kernels for second convPool layer, batch size at 40, poolsize is 2x2:

Table 2. Learning rate tuning table

Learning rate	0.1	0.05	0.01
validation error	97.5%	2.5%	5%
test error	N/A	5%	15%

From table 2 we can notice that learning rate of 0.05 is an optimized choice.

– Number of convolutional kernels in both LeNet Convolution Pool Layers: fixing batch size at 40, learning rate 0.05 and poolsize at 2 x 2:

Table 3. Number of kernels tuning table

n_kernel	[20,50]	[10,30]	[5,10]
validation error	2.5%	5%	5%
test error	5%	5%	7.5%

From table 3 we can see that optimal choice for number of convolutional kernels for the two conv layers is 20 and 50 respectively.

4 Discussion and Conclusion

4.1 PCA + SVM

For SVM part, the project reads face images as raw data and pre-process them by reducing dimensionality to decrease the load of training process and improve the efficiency of the classifier. Applying PCA as reduction dimensionality method is valid since it is fast as a preprocessing method and remain effective information from original data samples.

The SVM classifier is trained by RBF kernel function which is more flexible and more suitable for different sample data compared with linear and polynomial function. The reason for we choose RBF as the kernel function is that it can be applied to unlinear separated caseno matter how the dimensionality and sample is. For given different parameter γ and training data samples dimensionality, the accuracy could be different. From the testing part, we could draw a conclusion that the accuracy rely on remained dimensionality and parameter γ of classifier function seriously. The best accuracy for this dataset could be get when $\gamma = 0.0030$ while the dimensionality of generated data is 80, which is the highest one in our project, at some degree we can tell that the higher dimensionality could lead to better effect for classification.

4.2 CNN

Based on previous testing parameters, the final face recognition reults of the ORL database is mis-predicted 5 persons out of 400 images:

– picture: 89 is person 8, but mis-predicted as person 34

– picture: 178 is person 17, but mis-predicted as person 37

– picture: 189 is person 18, but mis-predicted as person 6

– picture: 299 is person 29, but mis-predicted as person 39

– picture: 368 is person 36, but mis-predicted as person 20

We use the red rectangular to point out these 5 picture positions as shown in Fig 5:

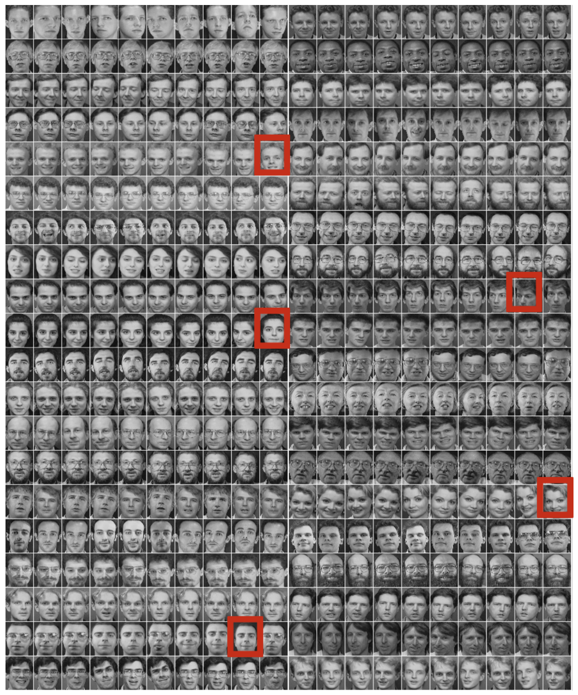

Fig. 5. Face Recognition Results by CNN Method

In conclusion, the success of the research in solving face recognition problem proves the feasibility of the proposed CNN model in the pattern recognition system.

4.2.1 Discussion and Future Improvement

- There are adaptive algorithms to determine the optimal learning rate. But due to the time constraint on this project, we manually tuned learning rate according to the validation and test error instead.
- Due to the limitation of the database, we set batch size as 40, which is the same size as the validation and test set. If the whole data set is larger we can avoid this and choose a smaller batch size and not making the error rate useless as it is here for batch size less than 40.
- The pool size parameter of the CNN model should also be tuned, however, the size of the sample data image is so small(57 x 47 pixels) that we don't have much space in designing the tuning set. If we have training data which has face images of higher resolutions, we could do similar tuning process for pool size. Note that source data of higher resolutions would cause the data

set size grow exponentially, which may need too much time to train on our own laptop.

References

1. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: Application to face recognition. IEEE transactions on pattern analysis and machine intelligence 28(12), 2037–2041 (2006)
2. Fasel, B.: Robust face analysis using convolutional neural networks 2, 40–43 (2002)
3. Heisele, B., Ho, P., Wu, J., Poggio, T.: Face recognition: component-based versus global approaches. Computer vision and image understanding 91(1), 6–21 (2003)
4. Lawrence, S., Giles, C.L., Tsoi, A.C., Back, A.D.: Face recognition: A convolutional neural-network approach. IEEE transactions on neural networks 8(1), 98–113 (1997)
5. LeCun Y., Bottou L., B.Y.H.P.: Gradient-based learning applied to document recognition. Proc. IEEE 86(11), 22782324 (1998)
6. Lyons, M., Akamatsu, S., Kamachi, M., Gyoba, J.: Coding facial expressions with gabor wavelets pp. 200–205 (1998)
7. Martínez, A.M., Kak, A.C.: Pca versus lda. IEEE transactions on pattern analysis and machine intelligence 23(2), 228–233 (2001)
8. Sun, Y., Wang, X., Tang, X.: Deep learning face representation from predicting 10,000 classes pp. 1891–1898 (2014)
9. Yang, M.H.: Face recognition using kernel methods. Advances in neural information processing systems 2, 1457–1464 (2002)