CMSC828C

Statistical Pattern Recognition

Project 1

Face Recognition

ADITYA VARADARAJ

(UID: 117054859)

(SECTION: 0101)

M.Eng. Robotics (PMRO),

University of Maryland - College Park, College Park, MD - 20742, USA.

Date of Submission: 14th November, 2022

List of Figures

List of Tables

CONTENTS

I. Problem Statement

The aim of this project is to implement different classifiers to achieve face recognition. A set of faces and their corresponding labels are given (3 datasets: **DATA, POSE, ILLUMINATION**). We first split the data into training and testing sets and use the training data to train the classifiers. First, dimensionality reduction was done using Principal Component Analysis (PCA) and Multiple Discriminant Analysis (MDA). The following classifiers were compared:

- Bayes Classifier
- k-Nearest Neighbors Classifier
- Kernel SVM with 2 kernels: RBF and Polynomial
- Boosted SVM (AdaBoost with Weak Linear SVM)

The 2 main classification tasks were:

- Identifying the subject label from a test image
- Neutral vs. Facial expression classification

II. Data Loading and Splitting

I performed both tasks **only** for the **DATA** dataset. The Train-test splitting and data loading had to be performed in 2 different ways depending on the classification task. Before splitting, we reshape the $24 \times 21 \times 600$ data to $504 \times 1 \times 600$.

A. Case 1: Subject Label recognition

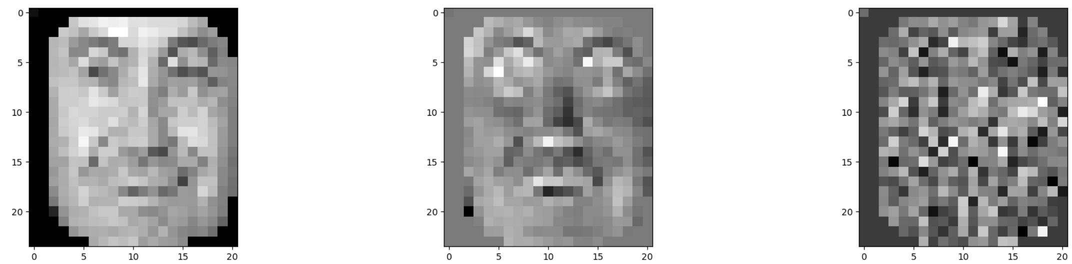
For this task, we label the data as 1,2,3,4,...,200 for the images corresponding to the 200 subjects, i.e., using subject no. as label. For this task, we need atleast 1 image per subject for all 200 subjects in both the training and test dataset. So we randomly choose 2 of the 3 images per subject to be put in training set. Thus, getting 400 images in training dataset and 200 in testing dataset.

B. Case 2: Neutral vs Facial Expression detection

For this task, we take only a subset of the data,i.e., we discard the third image per subject since the 1st image per subject is neutral, 2nd is facial expression, 3rd is only illumination changes. Thus, data becomes $504 \times 1 \times 400$. Now, we label the data as -1 or 1 depending on whether the face is neutral or having facial expression respectively. We need atleast 1 image of neutral and one of facial expression in both the training and test dataset. Do not necessarily need all subjects in training dataset. So we choose the first 300 images to be training dataset and rest 100 in testing dataset. Because of the way the data is structured, this means both neutral and expression images of 150 subjects in training and of 50 subjects in testing.

III. Preprocessing

This is done to reduce the dimensionality of the features of the data or to make data more seperable. The original image is shown in Fig. 1a



(a) (b) (c)

Fig. 1. (a) Original Image, (b) After PCA, (c) After MDA

A. PCA

Principal Component Analysis (PCA) projects data into a lower dimension to represent major aspects of data with minimum no. of features. After projecting the data, we reconstruct the higher dimension image from reduced dimension data. The output of PCA is shown in Fig. 1b. As can be seen, it preserves important features of the image.

B. MDA

MDA projects data into a lower dimension to get more seperable data. After projecting the data, we reconstruct the higher dimension image from the reduced dimension data. The output of MDA is shown in Fig. 1c.

IV. Classifiers

A. Bayes Classsifier

We first find Maximum Likelihood Estimates of priors, mean vectors and covariance matrices of the M classes as follows. Then we find the likelihoods and log of posterior for each class. We compare the log of the posterior, i.e., log of likelihood + log of prior for each class. The class for which the sample has maximum log of posterior is the predicted class for that sample.

B. kNN

We predict the label of the data based on the labels of the k nearest neighbors of the data. We experiment with value of k here.

C. Kernel SVM

We solve the Dual optimization problem

$$\max_{\underline{\mu}} \sum_{n=1}^{N} \mu_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \mu_n \mu_m y_n y_m K(\underline{x_m}, \underline{x_n}),\ subject\ to\ \underline{\mu} \geq 0 \tag{1}$$

$$\begin{gathered} f(.) = \sum_{n=1}^{N} \mu_n y_n K(., \underline{x_n}) \\ If\ \mu_k \neq 0,\ \theta_0 = y_k - f(\underline{x_k}),\ where,\ \underline{x_k}\ is\ a\ training\ sample \\ Predicted\ Label,\ y_{pred} = \theta_0 + f(\underline{x}),\ where,\ \underline{x}\ is\ a\ testing\ data\ point \end{gathered} \tag{2}$$

We use 2 types of Kernels:

- RBF Kernel $\left(K(x,y) = e^{\frac{-||x-y||^2}{\sigma^2}} \right)$ with hyperparameter σ
- Polynomial Kernel $(K(x,y) = (x^T y + 1)^r)$ with hyperparameter r

D. AdaBoost and Boosted SVM

A strong learner like SVM does not work well as the base learner of AdaBoost. However, a version of SVM that has been weakened can be useful [1]. The idea is to solve the optimization problem with a smaller training set, i.e., a subset of the original training set (50 out of the 300 images, i.e., 25 out of 150 subjects). We generate this subset randomly in each iteration of AdaBoost to get the weak classifier $\phi(\underline{x}_n, \theta_i)$ for that iteration of AdaBoost. Then we use AdaBoost algorithm:

Algorithm 1 AdaBoost Algorithm for Boosted SVM

$w_n^{(0)} \leftarrow 1,\ \forall\ n = 1, 2, .., N$
for $i = 1, 2, .., K$ **do**
  $P_n^{(i)} \leftarrow \frac{w_n^{(i)}}{\sum_{n=1}^{N} w_n^{(i)}}$
  $X_{sub}, y_{sub} \leftarrow \textsc{Random}(X_{train}, y_{train})$
  $\phi(x_n; \theta_i) \leftarrow \textsc{WeakLinearSVM}(X_{sub}, y_{sub})$
  $\varepsilon_i \leftarrow \sum_{n=1}^{N} P_n^{(i)} \mathbb{I}(y_n \neq \phi(x_n; \theta_i))$
  $a_i \leftarrow \frac{1}{2} \log \left(\frac{1 - \varepsilon_i}{\varepsilon_i} \right)$
  $w_n^{(i+1)} \leftarrow w_n^{(i)} e^{-a_i y_n \phi(x_n; \theta_i)}$

V. Results and Hyperparameter tuning

A. Task 1: Subject recognition

TABLE I
Results for Task 1: Subject Recognition

Classifier	Dimensionality Reduction	Hyperparameter ranges	Best hyperparameter values	Best Test Accuracy
Bayes	PCA			46.5
Bayes	MDA			60
kNN	PCA	$k = [1, 2, 3, 4]$	$k = 1$	40.5
kNN	MDA	$k = [1, 2, 3, 4]$	$k = 3$	60.5

1) Bayes Classifier: We can see that Bayes Classifier performs better in conjunction with MDA than PCA for the 1st task. From Table I, the accuracy of Bayes classifier with PCA is around 46.5% whereas with MDA is around 60%.

2) kNN: We can see that kNN Classifier performs better in conjunction with MDA than PCA for the 1st task. From Table I, the accuracy of kNN classifier with PCA is around 40.5% whereas with MDA is around 60.5%. We can see the best k is found to be $k = 1$ for PCA and $k = 3$ for MDA among choices $k = 1, 2, 3, 4$.

B. Task 2: Neutral v/s Facial Expression Classification

TABLE II
RESULTS FOR TASK 2: NEUTRAL V/S FACIAL EXPRESSION CLASSIFICATION

Classifier	Dimensionality Reduction	Kernel used	Hyperparameter ranges	Best hyperparameter values	Best Test Accuracy (%)
Bayes	PCA				85
Bayes	MDA				50
kNN	PCA		$k = [1, 2, 3, 4]$	$k = 2$	80
kNN	MDA		$k = [1, 2, 3, 4]$	$k = 1$	50
Kernel SVM	PCA	RBF	$\sigma = [10, 15, 20, 30]$	$\sigma = 10$	89
Kernel SVM	PCA	Polynomial	$r = [1, 2, 3]$	$r = 3$	89
Kernel SVM	MDA	RBF	$\sigma = [10, 15, 20, 30]$	No optimal solution found for Dual optimization problem	
Kernel SVM	MDA	Polynomial	$r = [1, 2, 3]$	$r = 1$	50
Boosted SVM	PCA		$K = [5, 10, 15, 20]$	$K = 15$	90
Boosted SVM	MDA		$K = [5, 10, 15, 20]$	No optimal solution found for Dual optimization problem in Linear SVM	

1) Bayes: We can see that Bayes Classifier performs better in conjunction with PCA than MDA for the 2nd task. From Table II, the accuracy of Bayes classifier with PCA is around 85% whereas with MDA is around 50%. 50% means all points are being classified in same class. So MDA doesn't work well for this classification problem.

2) kNN: From Table II, the accuracy of kNN classifier with PCA is around 80% with $k = 2$ whereas with MDA is around 50% with $k = 1$.

3) Kernel SVM: From Table II, the accuracy of Kernel SVM classifier with RBF kernel with PCA is around 89% with $\sigma = 10$ whereas with MDA has no optimal solution for the optimization problem. This again shows that MDA is unsuitable for this Neutral v/s Expression classification task.

From Table II, the accuracy of Kernel SVM classifier with Polynomial kernel with PCA is around 89% with $r = 3$ whereas with MDA is 50% with $r = 1$. This again shows that MDA is unsuitable for this Neutral v/s Expression classification task.

4) Boosted SVM: From Table II, the accuracy of Boosted SVM (AdaBoost with Weak Linear SVM) classifier with PCA is around 90% with $K = 15$ AdaBoost iterations whereas with MDA has no optimal solution for the Linear SVM optimization problem. This again shows that MDA is unsuitable for this Neutral v/s Expression classification task.

VI. Conclusion

- We conclude that Bayes and kNN Classifiers give better results with MDA rather than PCA in task 1 and with PCA rather than MDA in task 2.
- We conclude that both polynomial and RBF kernels give similar and good results in Kernel SVM with PCA with appropriate hyperparameters.
- We conclude that Boosted SVM gives the best accuracy among the classifiers used for task 2 with PCA.
- We conclude that MDA does not work well for classsification task 2, i.e., Neutral v/s Expression classification.

References

[1] García, Elkin and Fernando Lozano. “Boosting Support Vector Machines.”, 2007, MLDM Posters.