College of Design and Engineering
ELECTRICAL AND COMPUTER ENGINEERING

EE5907 Course Assignment 2

Face Recognition

A0280122W Ren Lingfeng
Personal Code Repository Link

2023/24 year

Contents

Chapter 1

Brief Introduction

1.1 Introduction

The selected topics are PCA, LDA, SVM, CNN, and GMM is also completed. Because notes are not noticed in the first place, I finished whole topics, and the results are provided. In this experiment, my own data is my personal photo shoot, which is also very different from the pie data, and its small number is also a large influencing factor in the later experiment.

Figure 1.1: own data visualization

The personal code repository will be released after the due date.
The link is there: Repository Link.

Chapter 2

Models

2.1 PCA

2.1.1 Experimental Results

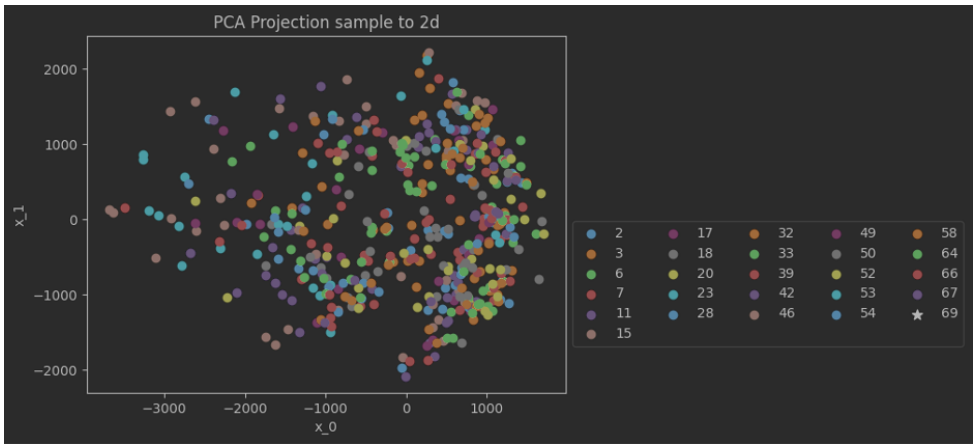

Figure 2.1: PCA 2d projection on samples

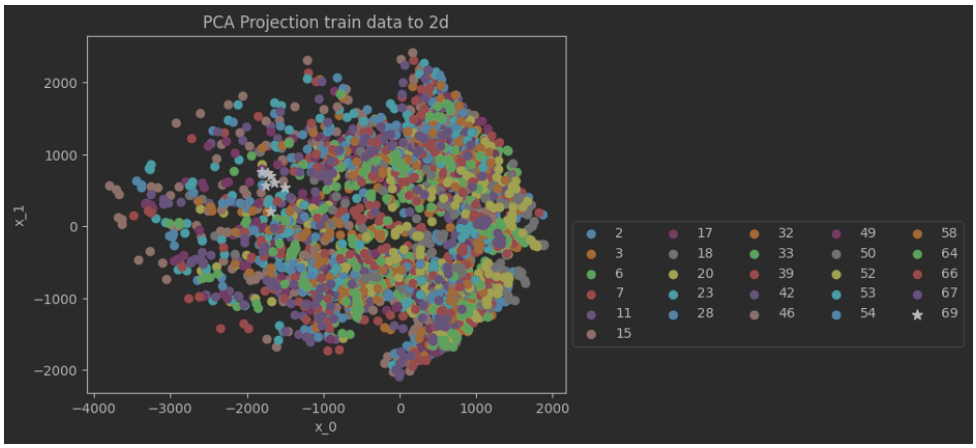

Figure 2.2: PCA 2d projection on train data

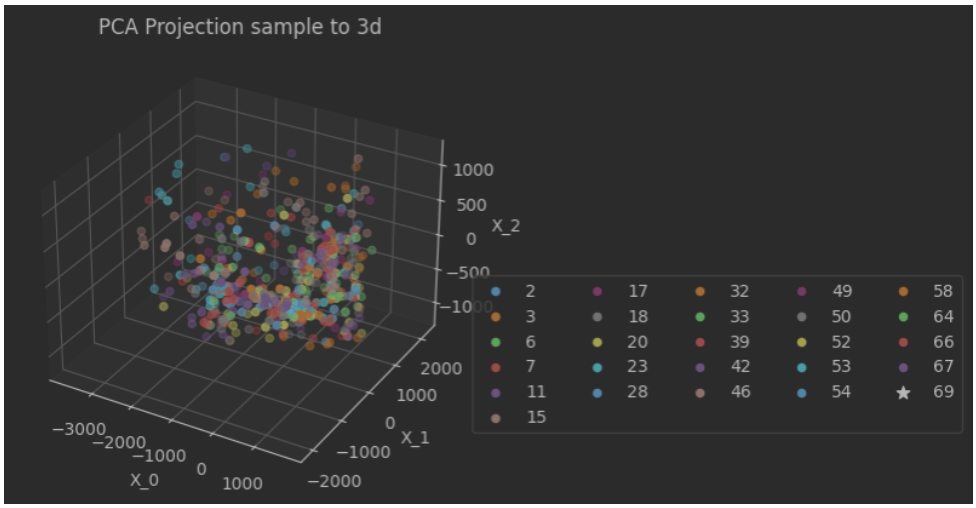

Figure 2.3: PCA 3d projection on samples

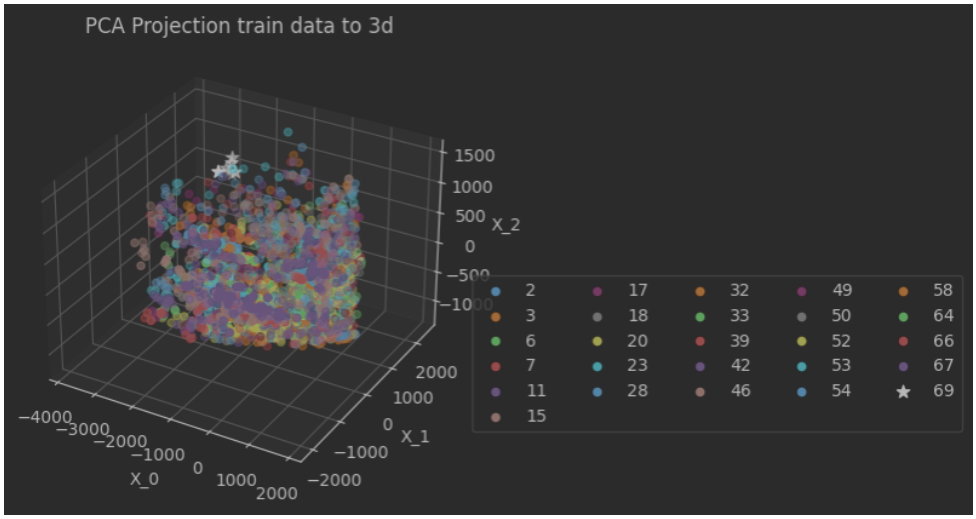

Figure 2.4: PCA 3d projection on train data

```
For pie data:
Accuracy -- dim 40 : 0.9450980392156862
Accuracy -- dim 80 : 0.9639215686274509
Accuracy -- dim 200 : 0.9709803921568627
For own data:
Accuracy -- dim 40 : 0.0
Accuracy -- dim 80 : 0.0
Accuracy -- dim 200 : 0.0
```

Figure 2.5: PCA plus nearest neighbor classification results

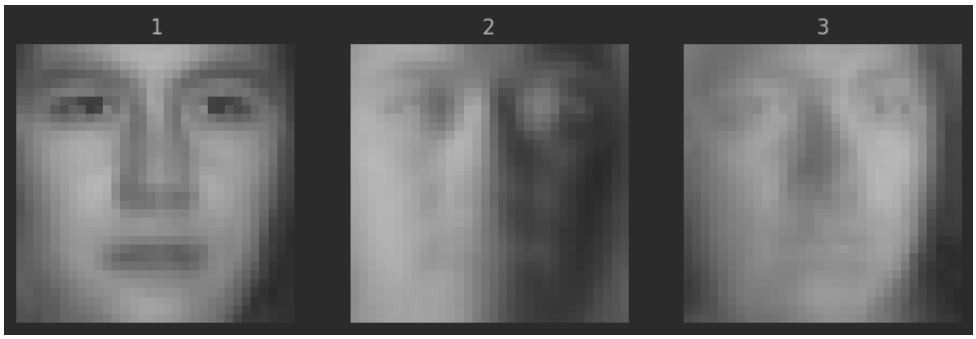

Figure 2.6: 3 eigenfaces

2.1.2 Analysis

It seems that own data is not selected in the random 500 samples, if you want to change the samples, you can change the variable 'seed', which ensure the experiment can be reproduced. the pie data is in one distribution, and it cannot be classified in 2 or 3 dimensions. According to figure 2.6, the faces can be reconstruct successfully, but there are some difference with the origin.

Figure 2.5 shows that own data cannot be classified, the reason is that own data has very different distribution compared with the pie data, the data is at the edge of the distribution, and this also can be seen in the next LDA part. And the pie data can be classified to some extent. The more dimensions are included, the higher the accuracy. Because the dimension reeducation means information loss, so an appropriate number of dimensions can balance the time consume and the accuracy.

Underrepresentation of data: If the PCA training sample contains no or very few photos of yourself, the generated principal components may not represent the features in your photos well. Overfitting: If PCA only learns a specific feature distribution from the original data set, it may overfit those features, and the model will perform poorly on any data that deviates from this distribution, such as my photos.

Class Imbalance: If there is a huge numerical imbalance between the original images and own photos in the data used for PCA training, PCA may tend to optimize for those data features that dominate the data and ignore the smaller data. feature. Significantly different features: Your photo may be very different from other images in the dataset in terms of lighting, angle, background, etc., causing PCA to be unable to capture features that are broadly representative of all data.

2.2 LDA

2.2.1 Experimental Results

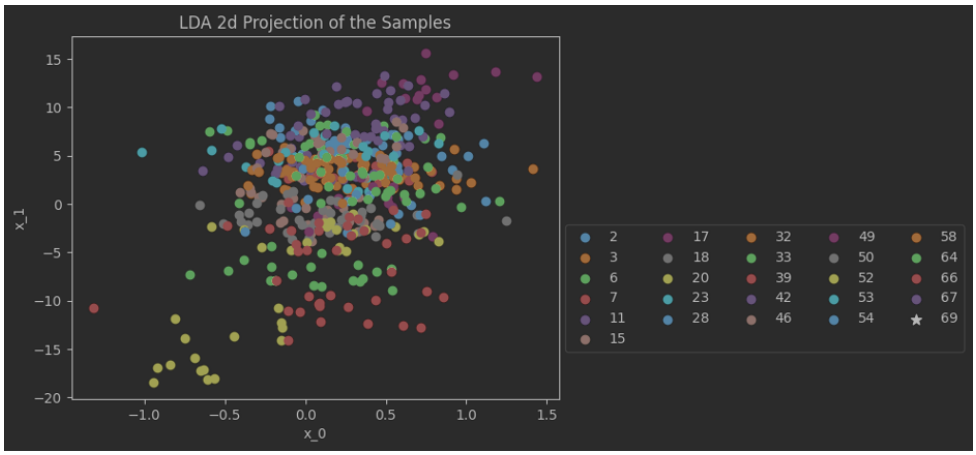

Figure 2.7: LDA 2d projection on samples

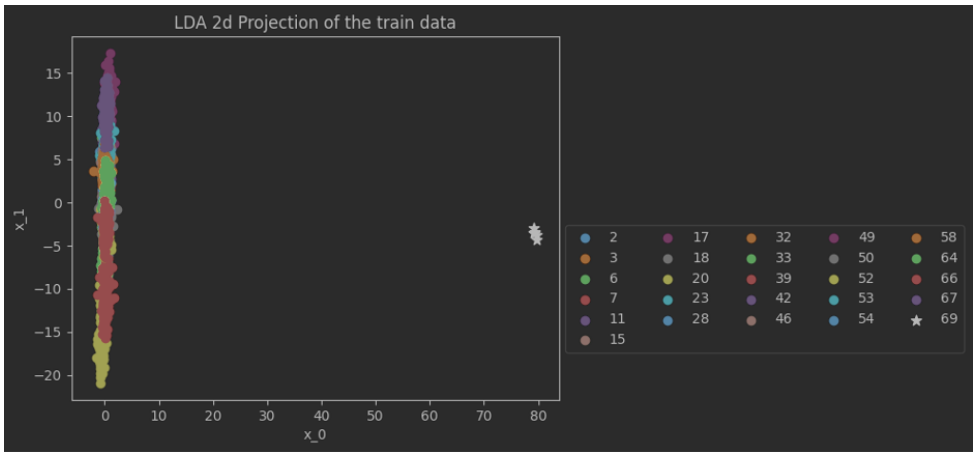

Figure 2.8: LDA 2d projection on train data

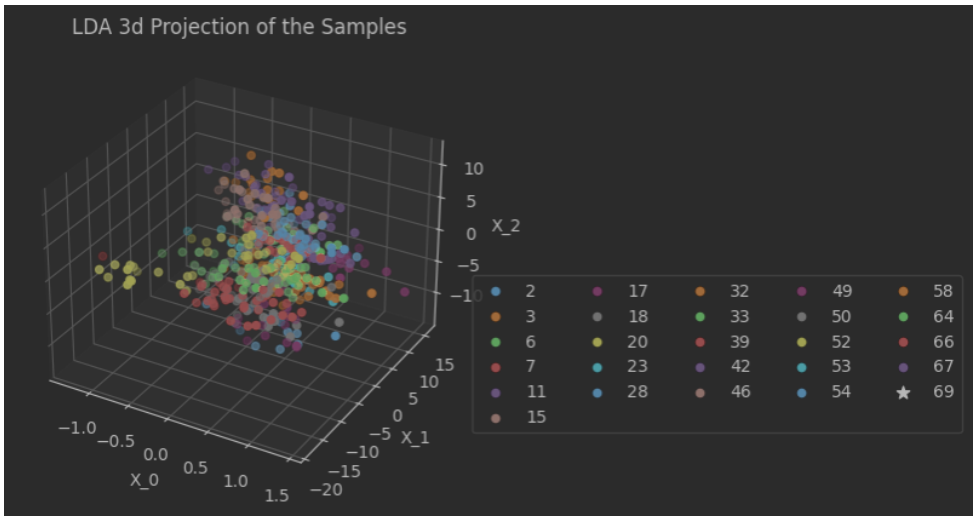

Figure 2.9: LDA 3d projection on samples

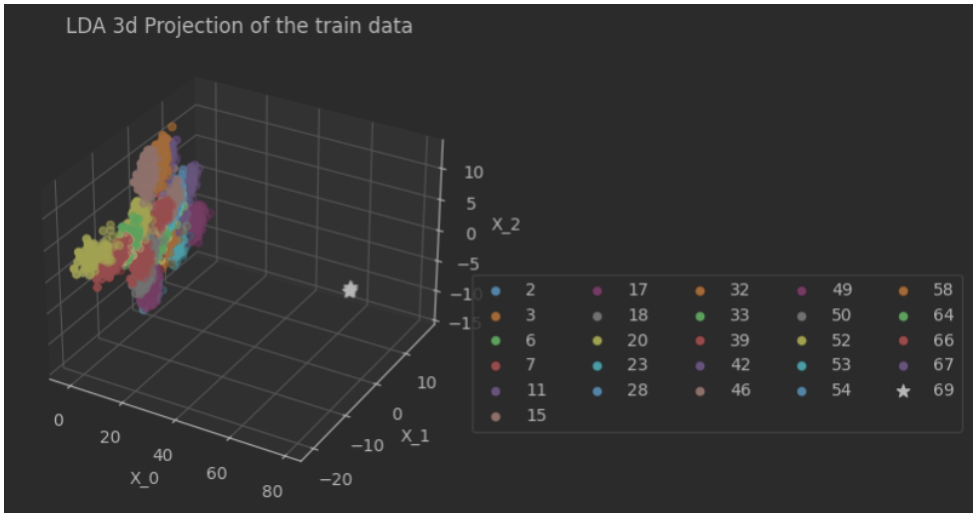

Figure 2.10: LDA 3d projection on train data

```
For pie data:
Accuracy of 2 = 0.23294117647058823
Accuracy of 3 = 0.4619607843137255
Accuracy of 9 = 0.9443137254901961
For own data:
Accuracy of 2 = 0.0
Accuracy of 3 = 0.0
Accuracy of 9 = 0.0
```

Figure 2.11: LDA plus nearest neighbor classification results

2.2.2 Analysis

Own data has very different distribution with the pie data in 2 or 3 dimensions. The categories in pie data are mixed, but they can be distinguished to a certain extent. 2 and 3 dimensions are not enough, and higher dimensions may be needed to obtain high accuracy.

Figure 2.11 showed that with higher dimension, model can gain higher classification accuracy. And own data is not selected in the random samples. However own data cannot be distinguished, and the reason is something like pca. Some reasons may cause this. Unbalanced sample size: If the number of samples in a category is much smaller than other categories, it may be "ignored" because it contributes less to the overall intra-class variance. Data premise hypothesis: LDA assumes that each category of data conforms to the multivariate normal distribution, and all categories have the same covariance matrix. If these assumptions do not hold, the performance of LDA may decline. Inter-class differences: LDA assumes that data of different categories have the same covariance structure. If your photo is substantially different in statistical properties from other data, LDA may not be able to find an appropriate linear combination to distinguish my photo from other categories of images. Training data representativeness: LDA requires sufficient samples per category to estimate intra-class covariance and inter-class covariance. If my photos are underrepresented in the training data, LDA may not learn correctly how to classify them. Class imbalance: Class imbalance may cause LDA to be too biased towards classes with large sample sizes. Make sure the training data is balanced across classes or use class weights. Overfitting: Although LDA is generally more resistant to overfitting than PCA, if my photo is fundamentally different from the images in the training set, the model may not classify my photo correctly regardless of the dimensionality reduction method used.

2.3 SVM

2.3.1 Experimental Results

```
setting: -t 0 -c 1 -q
Accuracy = 99.2175% (1268/1278) (classification)
Accuracy = 0% (0/3) (classification)
Accuracy of SVM with None-components for ALL test set: 99.218%
Accuracy of SVM with None-components for own test set: 0.0%
Accuracy = 99.1393% (1267/1278) (classification)
Accuracy = 100% (3/3) (classification)
Accuracy of SVM with 80-components for ALL test set: 99.139%
Accuracy of SVM with 80-components for own test set: 100.0%
Accuracy = 99.2958% (1269/1278) (classification)
Accuracy = 100% (3/3) (classification)
Accuracy of SVM with 200-components for ALL test set: 99.296%
Accuracy of SVM with 200-components for own test set: 100.0%
```

Figure 2.12: SVM setting: -t 0 -c 1 -q

```
setting: -t 0 -c 1e-1 -q
Accuracy = 99.2175% (1268/1278) (classification)
Accuracy = 0% (0/3) (classification)
Accuracy of SVM with None-components for ALL test set: 99.218%
Accuracy of SVM with None-components for own test set: 0.0%
Accuracy = 99.1393% (1267/1278) (classification)
Accuracy = 100% (3/3) (classification)
Accuracy of SVM with 80-components for ALL test set: 99.139%
Accuracy of SVM with 80-components for own test set: 100.0%
Accuracy = 99.2958% (1269/1278) (classification)
Accuracy = 100% (3/3) (classification)
Accuracy of SVM with 200-components for ALL test set: 99.296%
Accuracy of SVM with 200-components for own test set: 100.0%
```

Figure 2.13: SVM setting: -t 0 -c 1e-1 -q

```
setting: -t 0 -c 1e-2 -q
Accuracy = 99.2175% (1268/1278) (classification)
Accuracy = 0% (0/3) (classification)
Accuracy of SVM with None-components for ALL test set: 99.218%
Accuracy of SVM with None-components for own test set: 0.0%
Accuracy = 99.1393% (1267/1278) (classification)
Accuracy = 100% (3/3) (classification)
Accuracy of SVM with 80-components for ALL test set: 99.139%
Accuracy of SVM with 80-components for own test set: 100.0%
Accuracy = 99.2958% (1269/1278) (classification)
Accuracy = 100% (3/3) (classification)
Accuracy of SVM with 200-components for ALL test set: 99.296%
Accuracy of SVM with 200-components for own test set: 100.0%
```

Figure 2.14: SVM setting: -t 0 -c 1e-2 -q

```
setting: -t 0 -c 1e-8 -q
Accuracy = 39.2019% (501/1278) (classification)
Accuracy = 0% (0/3) (classification)
Accuracy of SVM with None-components for ALL test set: 39.202%
Accuracy of SVM with None-components for own test set: 0.0%
Accuracy = 35.759% (457/1278) (classification)
Accuracy = 0% (0/3) (classification)
Accuracy of SVM with 80-components for ALL test set: 35.759%
Accuracy of SVM with 80-components for own test set: 0.0%
Accuracy = 35.9937% (460/1278) (classification)
Accuracy = 0% (0/3) (classification)
Accuracy of SVM with 200-components for ALL test set: 35.994%
Accuracy of SVM with 200-components for own test set: 0.0%
```

Figure 2.15: SVM setting: -t 0 -c 1e-8 -q

2.3.2 Analysis

The accuracy is high and the remaining unclassified correctly may be mixed with other data, and the distribution is basically the same, resulting in a small change c that will not work.

The maximum tolerance has been reached: for some data sets, a specific C value may have minimized the number of misclassifications. Further adding a small amount of C will have no effect.

The range of change is not large enough: maybe the range of C changed is not large enough. For example, there may be no difference from 1 to 1e-2, but there may be significant changes from 1 to 1e-8.

In figure 2.16, the data is close to linear and divisible: if the data points are close to a linear decision boundary, small changes may not significantly affect the classification results.

Characteristics of data: In some cases, the decision boundary may be determined mainly by some key data points (or support vectors), and the location of these points may not change due to changes in C.

For dimension difference, smaller dimension may have lower accuracy than the original dimension, because it lost too much information, and reducing the dimension to an appropriate scale may has higher accuracy. In some cases, the reduced data may not provide a better effect than the original data. Here are some reasons that may cause this situation: Number of main components selected: Choosing too few main components may lead to information loss, while choosing too many main components may contain unnecessary noise. Usually, it is a good strategy to choose the right number of principal components based on the variance of interpretation. Information loss: dimension reduction means the loss of certain information. If the lost information is critical for tasks of concern (such as classification), then dimension reduction may lead to performance degradation.

2.4 CNN

2.4.1 Experimental Results

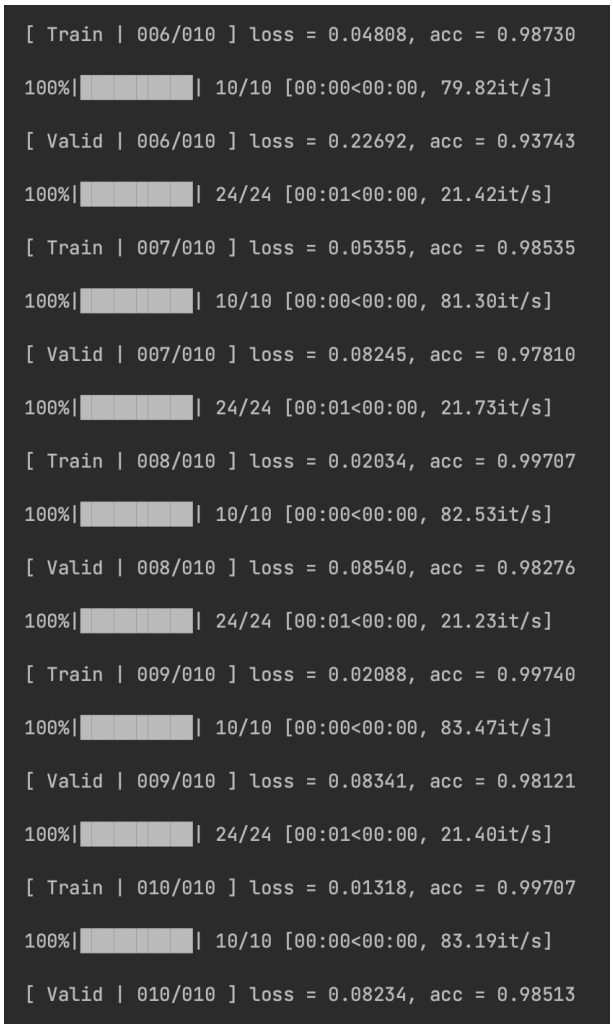

Figure 2.16: CNN: accuracy of last 5 epochs on train and test

2.4.2 Analysis

The model has very high accuracy on train data and test data in the final stage, after 10 epochs.

[Train | 010/010] loss = 0.01318, acc = 0.99707

[Valid | 010/010] loss = 0.08234, acc = 0.98513

The model performs well: from the perspective of absolute values, both loss values are very low, indicating that the model performs quite well in training data and validation data.

Low risk of overfitting: Generally, if the training loss is much lower than the validation loss, there may be an overfitting problem, that is, the model is too complex and perfectly fits every detail in the training data (including noise), resulting in poor performance on unseen data. But here, the training loss and the verification loss are very close, indicating that the risk of overfitting is small.

Model generalization ability: Because the validation loss is also low, this means that the model not only performs well in training data, but also performs well in validation data that it

has not seen. This is a good sign that the model has good generalization ability.

2.5 GMM

Additional section.

2.5.1 Experimental Results

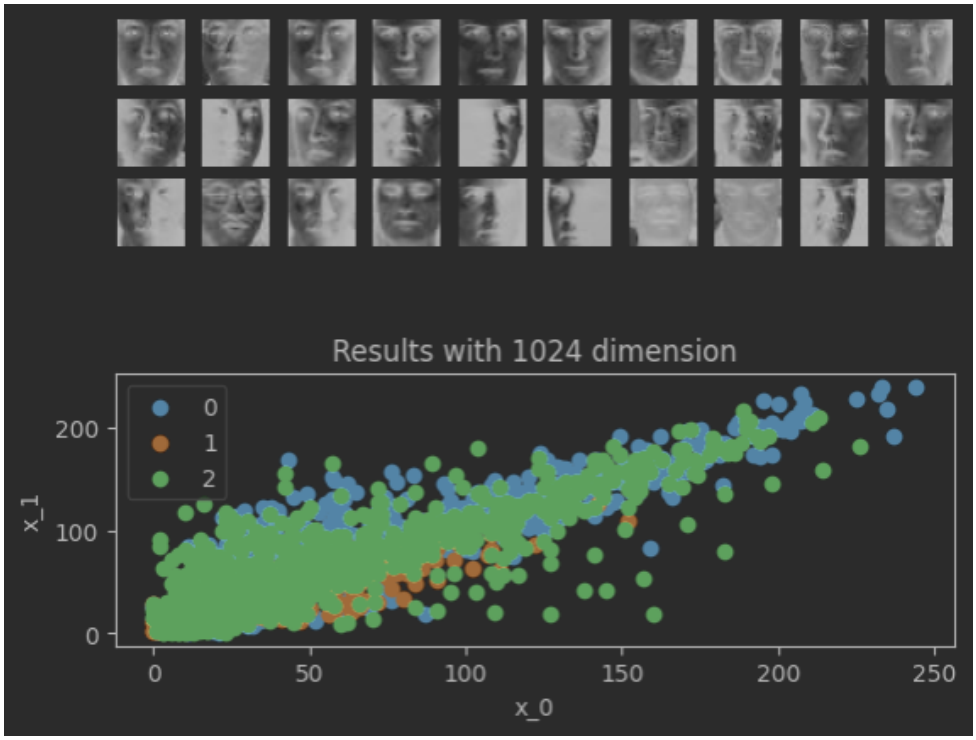

Figure 2.17: GMM in raw data

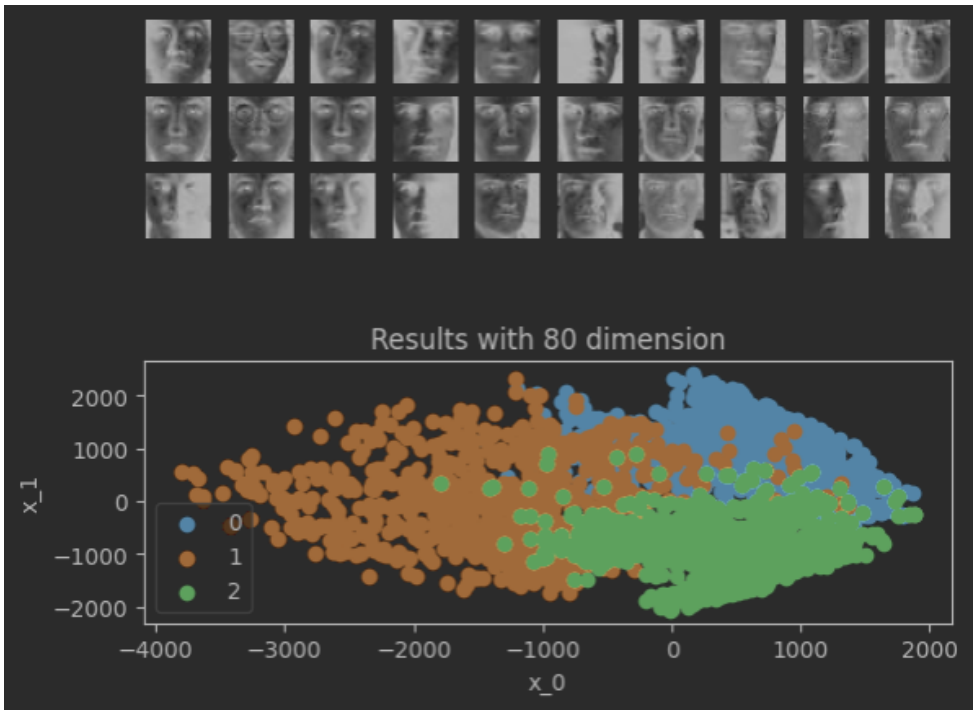

Figure 2.18: GMM in 80 dims

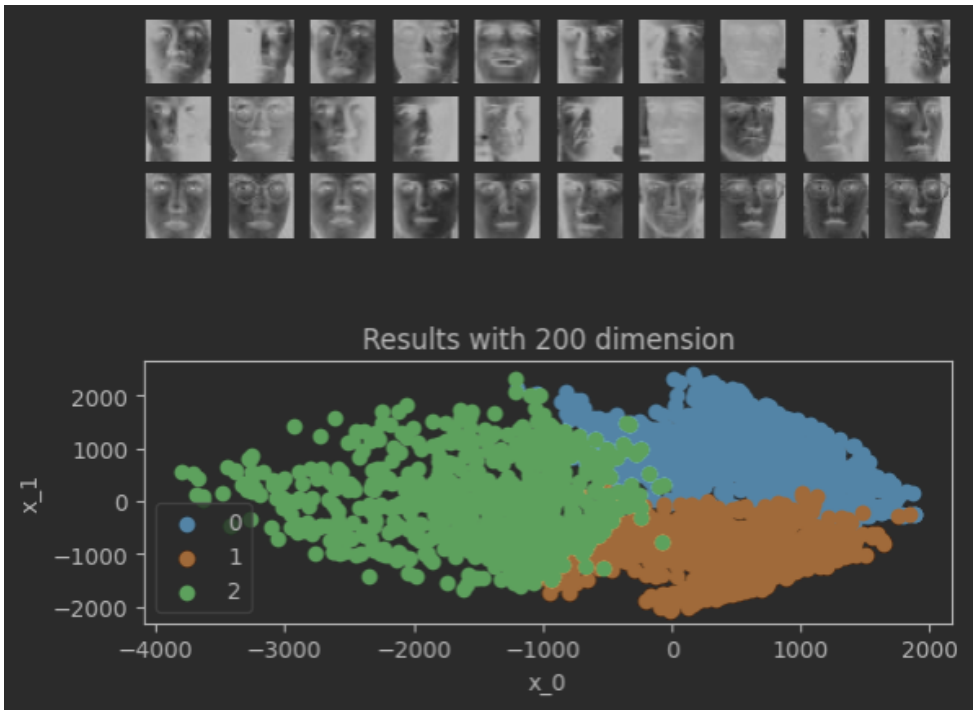

Figure 2.19: GMM in 200 dims

2.5.2 Analysis

With origin data, it cannot be distinguished, the distribution is mixed. With higher projected dimensions, the gmm can classify them with higher accuracy. The boundaries of data are clearer and there are fewer mixed numbers. Some possible reasons: Information preservation: Increasing the dimension of PCA means that more information about raw data is saved. GMM may benefit

from richer data, especially when the reduced dimensions lose information that is important for distinguishing each Gaussian component in the data.

Approaching nonlinear boundaries: Although GMM itself is linear, it can approximate nonlinear decision boundaries when you use multiple Gaussian components. In high-dimensional space, the structure of data may be more easily captured by multiple Gaussian components.

Complexity of data distribution: In some cases, the original data may contain multiple overlapping or close Gaussian distributions. Using high-dimensional PCA may be easier to distinguish these Gaussian distributions.

PCA variance retention: The purpose of PCA is to maximize the variance of data. When increasing the dimension, you will capture more variance, which may help GMM better understand the structure and distribution of data.

Reduction of noise and irrelevant features: In some cases, dimensionality reduction can help reduce noise or irrelevant features in the data. Although increasing the dimension of PCA will reduce this effect, in some cases, the moderate dimension may provide GMM with a good compromise between retaining critical information and noise reduction.

Chapter 3

Conclusion

3.1 Discussion

From the above models, we can see that neural network can gain the best results, but it also has maximum time consumption. A common approach is to use pca to get most useful dimensions, then use cnn, and it can reduce the time consumption.

A obvious evidence is that, the distribution of data can have a large impact on the model, and the unevenness of the categories can also have large influence.