**Harbin Institute of Technology (Shenzhen)**

**Image Processing Project Report**

**Experiment Name: Facial Expressions Recognition**

**Student Name: 许峻玮 陆琛**

**Student ID: 190110105 190110123**

**Report Date: 2021-11-08**

**Content**

[1 Project Content 1](#_Toc40128438)

[2 Method Description 1](#_Toc40128439)

[3 Experiment Results and Analysis 1](#_Toc40128440)

[4 Summary 1](#_Toc40128441)

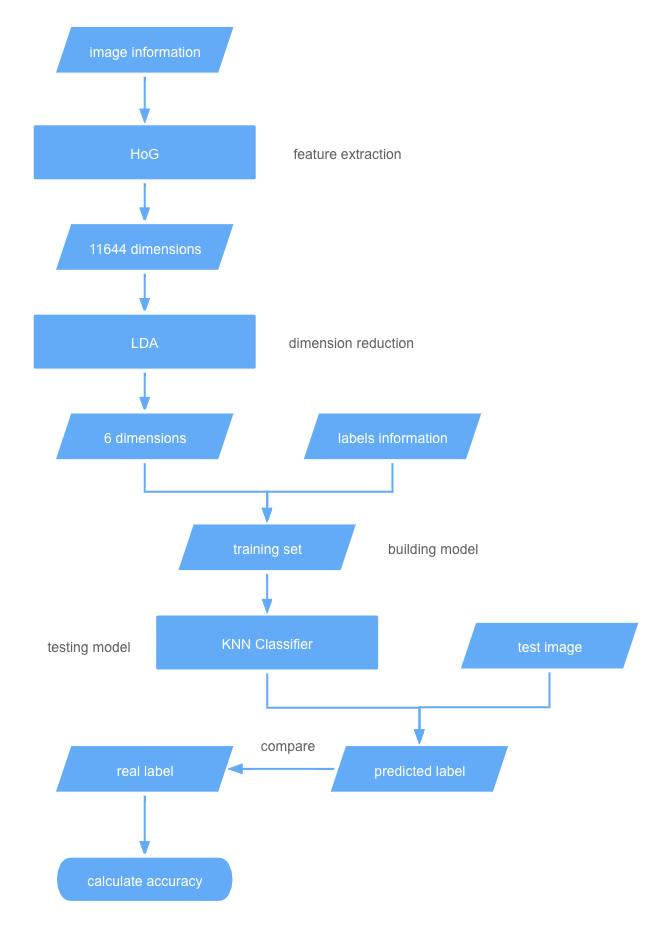
## 1 Project Content

In this experiment, we should try to judge a person’s facial expressions as precisely as possible, which could be happy, sad, surprise, fear, anger, disgust or neutral. To achieve this, we built a model that learned from known facial expressions. In other words, we told the computer this face looks sad while that seems fear. After training, we use another batch of faces to test whether the model can work well.

Hours of searching for methods and debugging, we finally made it and the accuracy of the model can be dramatically up to 88%, which is so satisfying.

## 2 Method Description

Algorithms used in this project are listed below.



It also reveals the process of our model.

**Step 1 Feature Extraction —— HOG**

The texture descriptor is widely used in computer vision for purposes such as detecti-ng objects. Usually, feature descriptors convert an image to an array vector by going over the image in patches of equal size. HOG as the name suggests provides a descriptor for the distribution of gradients in the image. This provides critical information about regions  of sufficient gradient changes like edges and corners. This helps increase the significance of objects in the paintings since those sudden changes provide more information about  object shapes.

This gradient is calculated for all fixed patches the image is divided into. Then a histogra-m is made out of the gradient direction based on the direction the gradient is in. We then normalize the histogram so that it doesn’t depend on lighting, shadows or contrast. In our implementation, we use the histogram of gradient implementation provided by sklearn.[[1]](#footnote-1)

**Step 2 Dimension Reduction —— LDA**

The basic idea of LDA is to project the high-dimensional pattern samples into the optimal discriminant vector space to achieve the effect of extracting classification information and compressing the dimension of feature space. After projection, it is ensured that the pattern samples have the maximum inter class distance and the minimum intra class distance in the new subspace, that is, the pattern has the best separability in this space.

**Step 3 Build and Test Model —— KNN**

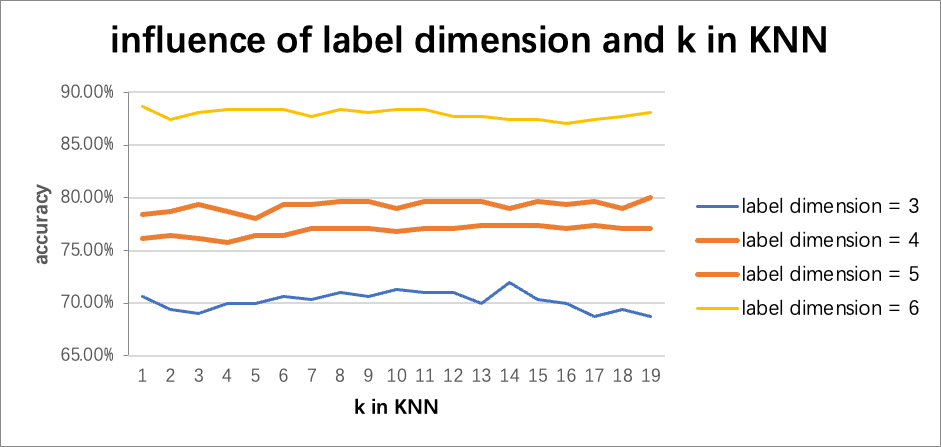
As the name suggests, this classifier considers the K nearest points in the neighbourhood of the given data point and classifies it into the majority class of those neighbours. The number of neighbours to be taken into account depends on the data set and results in varying levels of accuracy. In our implementation, we have taken KNN(5).

## 3 Experiment Results and Analysis

Generally, the accuracy of our model can always be over 65%. However, we believe it can be much higher after adjusting the arguments using in every algorithm.

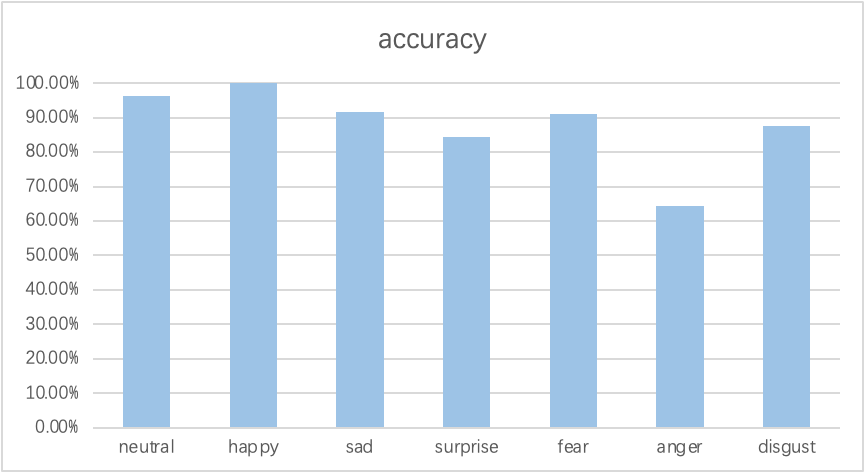
When we make the size of the window, block and cell smaller, reduce the label dimension to 6 and let k in KNN Algorithm be 5, we got an accuracy of 88.4%.

During adjustment, we keep recording our data and details are as following figures.

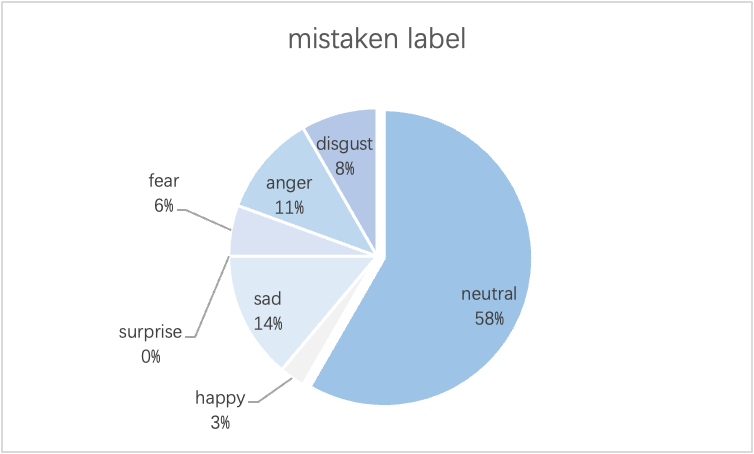


Picture above reveals that when we reduce label dimension to 6, the accuracy is much higher than others. Choosing different K value in KNN algorithm can also make a difference to the accuracy while fixing the label dimension. We notice that recommended K value is below 20, so we test the accuracy with different K value.

We are also interested in whether the accuracy is different from different label. Picture below shows every label’s accuracy and there comes a conclusion that our model can handle happy faces very well while a little hard on angry faces.



By the way, when a face is mistakenly recognized, we are eager to know which label did our model recognize. So here comes the third picture, which directly points out that in these mistakenly recognized faces, our model considers most of them as neutral faces.



## 4 Summary

1. What difficulties we encountered?

Having done project1, we’re more familiar with algorithms that are related to image process. So, it went pretty smoothly during this experiment. However, there is still something bothering us, one of which is adjusting the arguments. We met over 10 argument and the problem comes to be how to balance these arguments to get the highest accuracy. Easily but time-wasted, we use so many for-loop in order to explore better choice. Still, we cannot use for-loop while adjusting the size of the window, block and cell——We have to change little by little. And finally, we get the highest we could reach. Although there might be some other better combinations of the arguments.

1. What did we learn from this experiment?

We’re more proficient in DIP of course, especially using *Python* to achieve algorithms learned in class. Surprisingly, we also use algorithms learned in *Pattern Recognition*. Last but not least, our ability to coding and debugging enhanced a lot while facing and fighting with countless bugs, English writing is absolutely more fluent, too : )

This is the last experiment in DIP. But something really struck me while coding with DIP, making me couldn’t stop my pace on the way exploring the mystery of image process. Thanks a lot anyway and hope everything goes well 🙏🏻.

1. Aakansha Mathur,Shruti Arippara Vinodh,Siddhaling Urolagin. Classifying Paintings into Movements using HOG and LBP Features[A]. [↑](#footnote-ref-1)