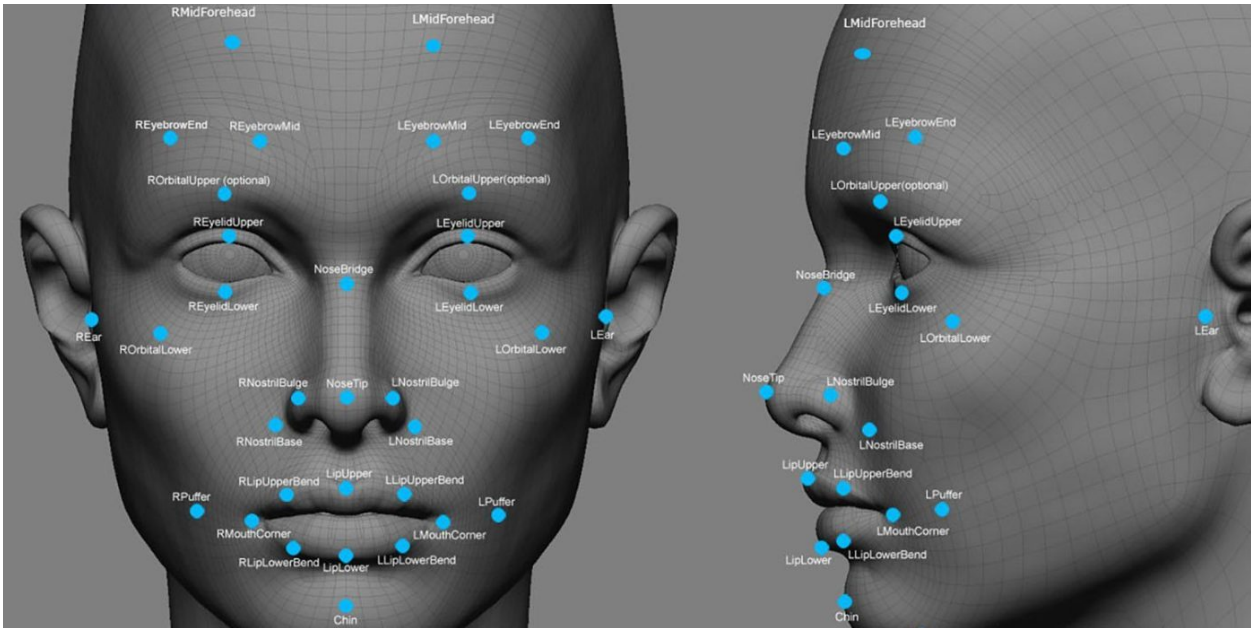
Facial Recognition and Machine Learning

Submitted to Prof. He, Xiaohai



Source: https://suppliertynews.com/2017/09/30/the-weekly-ai-report-the-bias-in-face-recognition-software/

Jian Gao

@ Sichuan University

August 6, 2019

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

GLOSSARY

1.0 INTRODUCTION

2.0 FACIAL ATTRIBUTES AND CATEGORIZATION

- **2.1 Common Attributes**
- **2.2 Additional Features**

3.0 FACIAL RECOGNITION USING A CONVOLUTED NEURAL NETWORK (CNN)

- **3.1 SIFT Algorithm (Phase 1 & 2)**
- **3.2 MNIST Database (Phase 3)**
- **3.3 Training & Testing (Phase 4)**

REFERENCES

RESOURCES

LIST OF ILLUSTRATIONS

Figure 1. Facial Points Locations

Figure 2. Hairline Shapes

Figure 3. Six Commonly Seen Types of Eyes

Figure 4. Dog in SIFT

Figure 5. John Lennon with SIFT

Figure 6. Neural Network Layout

Figure 7. Neural Network Implementation

Figure 8. The Original, Blurred, and Edge-detected

Figure 9. Training Process

Table 1. Nasal Index (Percentage)

Table 2. Facial Attributes and Their Types

Table 3. Additional Features

Table 4. Result_1.csv

Table 5. Result_2.csv

GLOSSARY

OpenCV	Open Source Computer Vision Library
SIFT	Scale-invariant feature transform
DoG	Difference of Gaussians
MNIST	Mixed National Institute of Standards and Technology
Gradient Descent	An optimization algorithm for finding the minimum of a function

1.0 INTRODUCTION

As an essential part of human bodies, human faces vary in several ways. In the scope of anthropology, this may be caused by different food, water sources, climates, languages during the march of evolution, and even social achievement/economic growth in modern days (Mehrotra 1997). According to the researchers at the University of California (Berkeley), the variation among faces is quite significant comparing humans to most of the other creatures.

It might be obvious to us that interattribute distances (e.g., mouth-nose distance, interpupillary distance) matter the most when identifying a person, while little evidence stands for this claim (Dupuis-Roy 2014). In contrast, according to the phycological research carried out in Canada and Britain, attribute shapes and skin properties help with the recognition process significantly.

Additionally, because many human face databases do not include subjects' ears in their models, we neglected them along with interattribute distances. Thus, in this paper, a human face is studied as the combination of five major segments which are a forehead, eyes, ears, a nose, and a mouth (lips). Generally, at least 50 points can be extracted from the source image as Facial Points (Figure 1) outlining the features on a face. Then, these points are joint by B-spline method to reconstruct the shapes for further analysis (Katsikitis 2003).

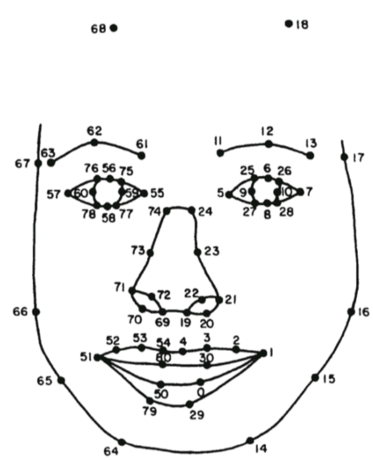


Figure 1. Facial Points Locations

2.0 FACIAL ATTRIBUTES AND CATEGORIZATION

2.1 Common Attributes

Diving into the topic, let's look at the divisions of the forehead at first. Two shapes are commonly seen—curved foreheads, which look protrudes and full from the side, and slopped ones on the contrary. Additionally, hairline types and forehead lines (Figure 2) also have their distinguishable characteristics.

Figure 2. Hairline Shapes

Coming down to the eyes and eyebrows, attached is a figure that helps categorization. Interestingly, it is said that eyeballs and eyelids can be great indicators of age since many changes and diseases of eyes are strongly age-related (Cavallotti 2008), such as cataracts, glaucoma, and diabetic retinopathy.

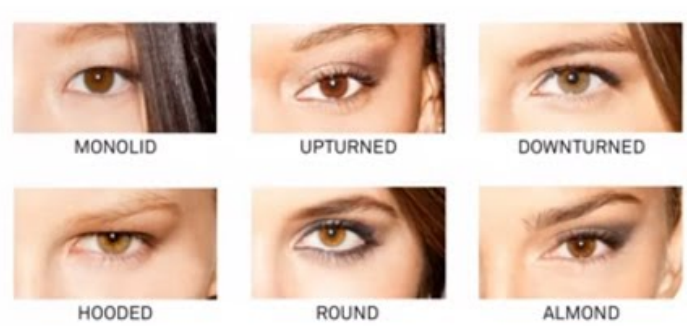


Figure 3. Six Commonly Seen Types of Eyes

As for noses, we tend not to worry about the actual shapes that much. Instead, a new concept gets introduced—quotient indices (Katsikitis 2003) which express one facial dimension as a percentage of the other. As the classification scheme (Table 1) illustrates, the width of a nose to its height usually ranges from 55% to 100%.

Quotient Indices for Noses	
Very narrow	X-54.9
Narrow	55.0-69.9
Medium	70.0-84.9
Wide	85.0-99.9
Very Wide	100.0-X

Table 1. Nasal Index (Percentage)
Source: Katsikitis (2003)

Lastly, lips can be any of the three types—Thin, Normal, or Thick. Moving back to the broader perspective, skin tones and face shape can also add to the complexity of a face (Alashkar 2017). To include all the facial characteristics we have studied, a complete table of our model is now concluded (Table 2).

Facial Attribute	Types
Hairline	Uneven, M-shaped, Widows' Peak
Forehead	Curved, Slopped
Eyes	Monolid, Upturned, Downturned, Hooded, Round, Almond
Nose	Narrow, Medium, Wide
Lips	Thin, Normal, Thick
Skin Color	Light, Fair, Medium, Black
Face Shape	Oval, Square, Round

Table 2. Facial Attributes and Their Types

2.2 Additional Features

Upon the request of developing a program that generates faces according to the input from the operator, additional features are also taken into consideration since we are seeking for more detailed descriptions. In this case, the features are represented as binary values with 1 for being present and 0 on the contrary (Zhang et al. 2015). Specifically, in regard of gender, 1 stands for male and 0 for female. Other attributes may be added for research purposes as well.

Group	Attributes
eyes	bushy eyebrows, arched eyebrows, narrow eyes, bags under eyes, eyeglasses
nose	big nose, pointy nose
mouth	mouth slightly open, no beard, smiling, big lips, mustache
global	gender, oval face, attractive, heavy makeup, chubby
head pose	frontal, left, left profile, right, right profile

Table 3. Additional Features

Source: Zhang et al. (2015)

3.0 FACIAL RECOGNITION

USING A CONVOLUTED NEURAL NETWORK (CNN)

This section involves the programming language of Python, TensorFlow, and Open Source Computer Vision Library (OpenCV). For the purpose of keeping the codes light and clean, I used Anaconda-Navigator on macOS as suggested in *Practical Machine Learning and Image Processing* (Singh 2019). After installing the related packages, our journey begins.

3.1 SIFT Algorithm (Phase 1 & 2)

Available in OpenCV, scale-invariant feature transform (SIFT) is a rather easy and useful algorithm for identifying the similarities between images. Based on my own understanding, SIFT can be derived as a 3-step process. First of all, to make the comparison scale-irrelevant, the algorithm scales both images to various sizes and generates sets of gaussians respectively, after which Difference of Gaussian (DoG) is obtained in each set. Then, we search for local extrema that are seen as potential key points, and the points with lower intensity (a.k.a. unimportant key points) are neglected. Within the 16x16 pixel block around a key point, the orientation of a key feature is captured and sent to match the counterpart from the other image, yet they should be not exactly the same.

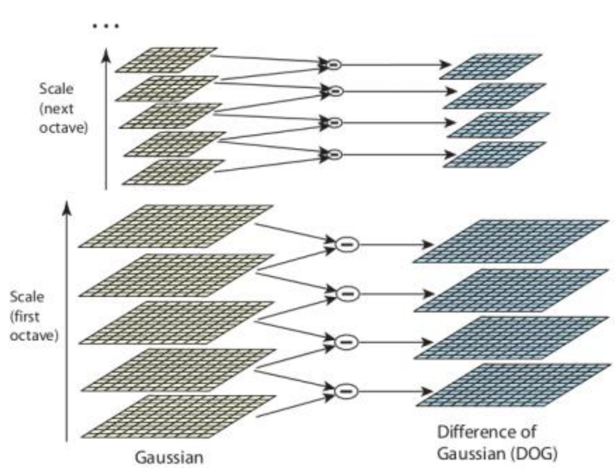


Figure 4. DoG in SIFT

Source: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_sift_intro/py_sift_intro.html

Here is the result (Figure 5) of a test run using a John Lennon's portraits, and its code can be found in Resources. There is a noticeable amount of mismatches, but it provides us with a good sense of what this patented algorithm does.

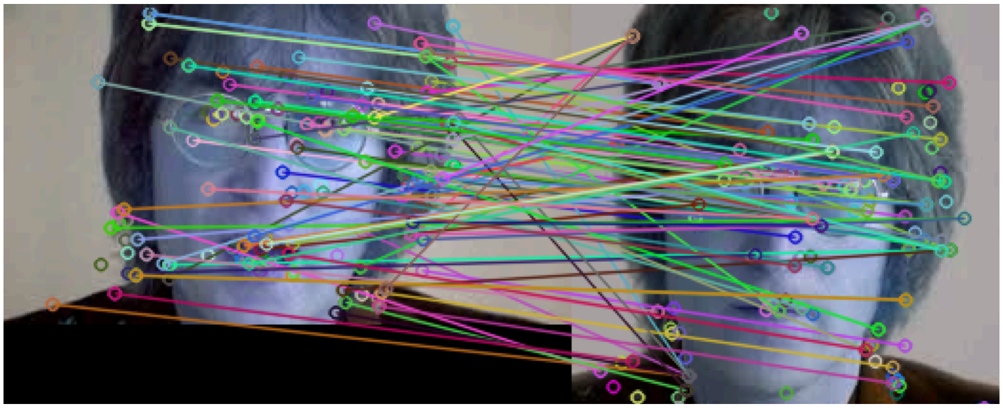


Figure 5. John Lennon with SIFT

3.2 MNIST database (Phase 3)

Before creating and training our own neural network for facial recognition, let us get to know about MNIST dataset. It's a csv file that consists of 785 columns indicating the label and the pixel values extracted from 28x28 handwritten single digit images. We use Python to read the file and input it as the training set to a 4-layer neural network. After a few minutes, our network is ready for the tests.

3.3 Training & Testing (Phase 4)

To carry it on, our goal is now identify a person from an image among a group of people. Thus, we downloaded a set of portraits of 40 people under different lightning circumstances from Extended Yale Face Database B. Given that the pictures (in pgm format) for an individual are enclosed in one folder, we target our program to the folders. While iterating through them, a single csv file with pixel values is now compiled. For this, we implement a similar structure of neural network (Figure 6) as in MNIST.

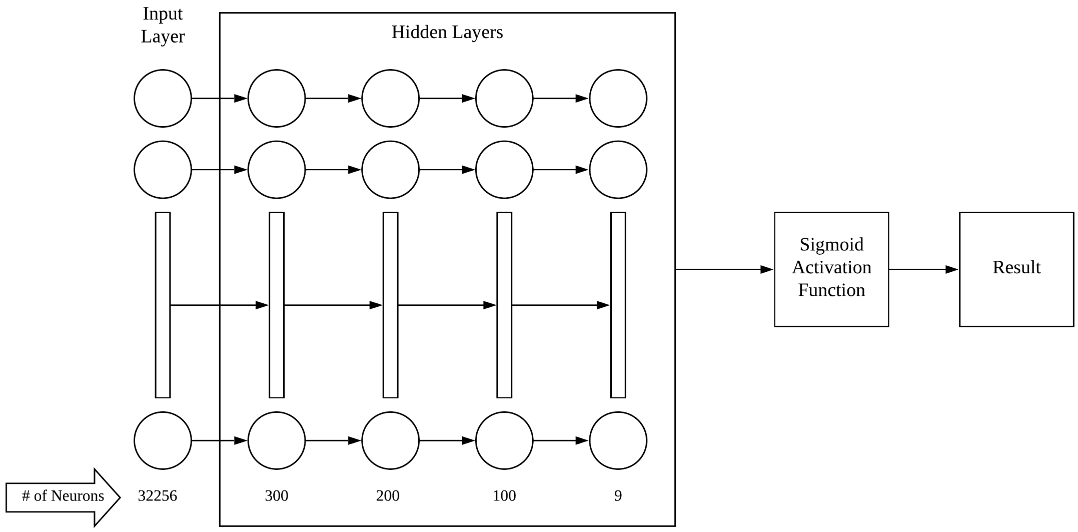


Figure 6. Neural Network Layout

```
classifier = Sequential()
classifier.add(Dense(units = 300, kernel_initializer ='uniform', activation = 'relu', input_dim = 32256))
classifier.add(Dense(units = 200, kernel_initializer ='uniform', activation = 'relu'))
classifier.add(Dense(units = 100, kernel_initializer ='uniform', activation = 'relu'))
classifier.add(Dense(units = 9, kernel_initializer ='uniform', activation = 'sigmoid'))
classifier.compile(optimizer = 'sgd', loss = 'mean_squared_error', metrics = ['accuracy'])
classifier.fit(X, y, batch_size = 10, epochs = 30)
```

Figure 7. Neural Network Implementation

However, prior to plug the pixels values in the chart to input layer, a Gaussian blur filter is needed to be applied to the source images to reduce the noise as much as possible. Additionally, we call cv2.Canny to enable canny edge detection included in OpenCV so that only the outline of face and facial attributes gets thrown into the network from which we expect the best result. Besides, gradient descent method is carried out before the actual training process starts.

Figure 8. The Original, Blurred, and Edge-detected

```
Epoch 21/30
583/583 [==============================] - 4s 6ms/step - loss: 0.0341 - acc: 0.7616
Epoch 22/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0325 - acc: 0.7719
Epoch 23/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0310 - acc: 0.7787
Epoch 24/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0296 - acc: 0.7873
Epoch 25/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0287 - acc: 0.7925
Epoch 26/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0281 - acc: 0.7942
Epoch 27/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0275 - acc: 0.8062
Epoch 28/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0272 - acc: 0.8079
Epoch 29/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0264 - acc: 0.8165
Epoch 30/30
583/583 [==============================] - 3s 6ms/step - loss: 0.0255 - acc: 0.8216
```

Figure 9. Training Process

As Python rolls out epochs, accuracy shows a significant upward trend with a continuous drop in loss. With an accuracy at 0.8216, we have tested five images from subject No. 4, 5, 7, 9, and 15 respectively, while the files are taken out of our training set in advance. The code that extracts the predicting values is given in **RESOURCES**. The figures in Table 4 demonstrate the possibility that the image might belong to the illustrated group. It is noted that the third test image is tagged as No.4 which is incorrect due to the uncertainty in our neural network.

1	2	3	4	5	6	7	8	9	True/False
9.2e-04	8.1e-04	1.6e-02	1.0e+00	1.0e-06	1.2e-04	8.0e-03	9.7e-05	4.3e-05	T
1.9e-06	3.2e-05	4.0e-02	3.8e-05	9.4e-01	3.5e-04	8.7e-03	0.0e+00	2.0e-02	T
7.6e-04	9.9e-05	2.2e-06	2.3e-01	3.0e-04	1.4e-05	1.4e-01	2.2e-04	1.2e-05	F
0.0e+00	1.8e-07	3.3e-06	1.8e-05	0.0e+00	8.1e-03	2.2e-04	6.6e-07	1.0e+00	T
3.7e-03	1.4e-02	7.2e-05	3.2e-01	4.8e-03	9.9e-04	1.9e-02	1.3e-05	9.8e-05	T

Table 4. Result_1.csv

To boost the accuracy of prediction, we launch 60 of epochs at a time. As it is shown (Table 5), result comes out as expected.

1	2	3	4	5	6	7	8	9	True/False
9.2e-04	2.1e-02	1.2e-01	9.6e-01	2.0e-01	6.8e-02	7.0e-02	4.3e-05	1.0e-06	T
8.0e-02	2.9e-02	0.0e+00	8.0e-02	1.0e+00	3.0e-02	3.2e-05	9.0e-02	3.8e-05	T
1.8e-07	8.0e-07	6.1e-04	3.0e-06	1.5e-03	1.9e-05	6.1e-01	1.1e-01	1.2e-05	T
5.1e-06	0.0e+00	6.8e-04	6.0e-08	1.6e-01	1.9e-06	1.8e-02	0.0e+00	9.1e-01	T
1.5e-02	1.1e-02	2.9e-02	1.0e-02	9.8e-05	2.4e-02	4.3e-03	2.0e-02	3.1e-01	T

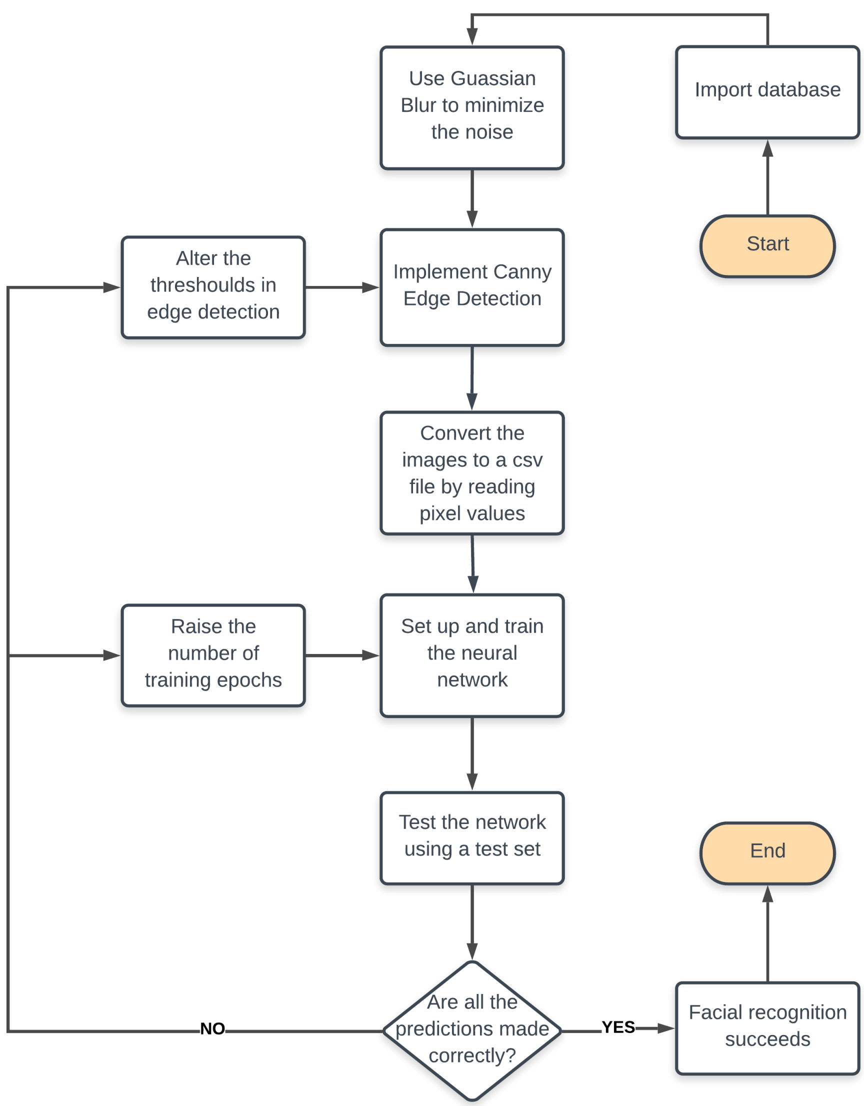
Table 5. Result_2.csv

REFERENCES

1. https://suppliertynews.com/2017/09/30/the-weekly-ai-report-the-bias-in-face-recognition-software/
2. https://www.police.qld.gov.au/programs/cscp/Documents/PersonDescriptionFactSheet.pdf
3. https://www.yourchineseastrology.com/face-reading/forehead/
4. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_sift_intro/py_sift_intro.html
5. Mehrotra, S. K., & Jolly, R. (1997). *Development with a human face: Experiences in social achievement and economic growth*. New York; Oxford; Clarendon Press.
6. Dupuis-Roy, N., Fiset, D., Dufresne, K., Caplette, L., & Gosselin, F. (2014). *Real-world interattribute distances lead to inefficient face gender categorization*. Journal of Experimental Psychology: Human Perception and Performance, 40(4), 1289–1294. https://doi-org.ezproxy.library.ubc.ca/10.1037/a0037066.supp
7. Vezzetti, E., Marcolin, F. (2012). *3D human face description: landmarks measures and geometrical features*, Image and Vision Computing, Volume 30, Issue 10, Pages 698-712, ISSN 0262-8856
8. Cavallotti, C. (2008). *Age-related changes of the human eye*, ISBN: 1-934115-55X, 978-1-934115-55-8, Humana Press
9. Katsikitis, M. (2012). *The Human Face: Measurement and Meaning*, ISBN: 1-4613-5376-9, 978-1-4613-5376-8, Springer
10. Alashkar, T. & Jiang, S. & Wang, S. & Fu, Y. (2017). *Examples-Rules Guided Deep Neural Network for Makeup Recommendation.*
11. Zhang, Z. & Luo, P. & Loy, C. C. & Tang, X. (2015). *Learning Deep Representation for Face Alignment with Auxiliary Attributes*. IEEE Transactions on Pattern Analysis and Machine Intelligence. 38. 1-1. 10.1109/TPAMI.2015.2469286.
12. Singh, H., SpringerLINK ebooks - Professional and Applied Computing, SpringerLink (Online service), & O'Reilly for Higher Education (2019). *Practical machine learning and image processing for facial recognition, object detection, and pattern recognition using python* (1st ed.). Berkeley, CA: Apress.

RESOURCES

1. OpenCV-Python Tutorial: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_tutorials.html
2. MNIST Dataset: https://www. kaggle.com/c/digit-recognizer/data#
3. Extended Yale Face Database B: http://vision.ucsd.edu/content/extended-yale-face-database-b-b
4. Diagram Drawing Tool: https://lucid chart.com
5. Python Problem Solving: https://stackoverflow.com
6. My code for section **3.0**: https://github.com/jian-99/Code_Online
7. Flowchart for section **3.0**:



Special Acknowledgements:

Ms. Lan, Jie

Prof. He, Xiaohai

@Sichuan University