**Detecting Sentiment in Human Faces**

CIS 678

Jie Tao

Joel Smith

**Background**

The purpose of this project is to use supervised learning to detect sentiment in human faces. This could be used for smart human-computer interaction, such as lie detection, or adding an emotion tag to a movie or video. Most of the research we found in this area uses deep learning, especially the most recent work. The newest research paper we found on the topic that does not use deep learning is from 2008. Our plan was to not use deep learning and see how well our model performs compared to state of the art deep learning models.

According to the trained model, we can detect 7 different emotions (0=anger, 1=contempt, 2=disgust, 3=fear, 4=happy, 5=sadness, 6=surprise) according to any given frontal view. In this project, we excluded 0 - neutral.

Three major parts of this project are human face detection, feature extraction and model design. Human face detection is finding the position of face in a given picture; feature extraction is reducing the noise in the cropped face picture and keeping major features; model design is discovering a proper model, training and evaluating it. The packages or technologies facilitating this project are listed in **Table 1**.

|  |  |  |
| --- | --- | --- |
| **Section** | **Package** | **Description** |
| Data Preparation | oversample  sklearn | Frame forward  StratifiedShuffleSplit |
| Face detection | OpenCV  dlib | Facedetect  Draw face basic features |
| Feature extraction | SIFT  Gradient  Canny | Difference of Gaussian  Gradient by orientations  Outline of objects |
| Modeling | sklearn | SVM  MLP  PCA |

Table 1 - a brief description of packages or technologies in sentiment detection

The datasource is from **Extended Cohn-Kanade (CK+) database**. All sequences are from the neutral face to the peak expression. Thus, it captures 593 peak expression frame across 123 subjects, and classify to 8 emotions. See **Figure 1**. But only 327 of the 593 sequences have marked classification labels. This is because these are the only ones the fit the prototypic definition.

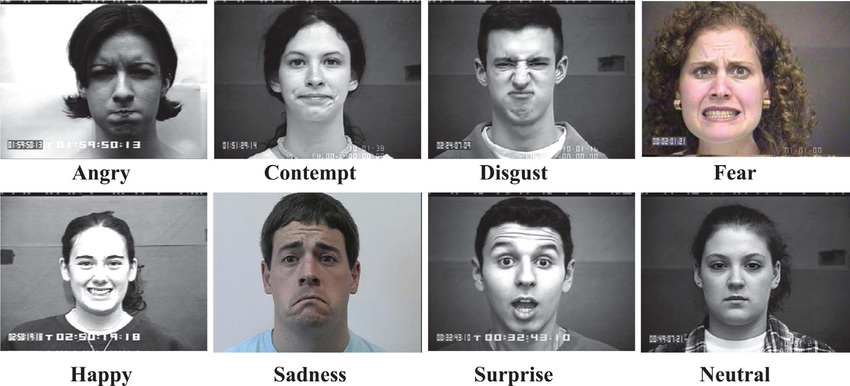


Figure 1 - eight different emotions in the training samples, which are all in the peak expressional frame

**Algorithms Used**

1. **Human face detection**

A trained human face library in OpenCV has been used to find the position and width and height of the face in the samples. To more focus on face, we squeeze 15% size of width and height respectively, then compress to 200\*200 pixels gray picture( the color channels are useless for sentiment detection). According to **Figure 2**, we can find the major expressions have been captured, and cropped ears and mostly hair out of the frame.



Figure 2 - face detection and crop in the gray channel

Before stepping into feature extraction, another more accurate face detection called **dlib,** has been used to find detail 68 points on the face, such as eyebrow, eyes, nose, mouth, nose, and jawline. In this project, we connected all points except ones on nose, inner mouth and jawline. **Figure 3**.



Figure 3 - with connected points, we can find different shapes under different emotions.

1. **Feature extraction**

After the face detection, we tried to do the feature extraction with three different ways:

* **SIFT**

SIFT use difference of Gaussian to detect the object features. We can find the small key points has capturing some important information, such as frown and edge of nose. See **Figure 4**. Each keypoint consists of descriptor array with 128 length. Since the number of keypoints in each image are different, we use k-mean to reduce number of keypoints to 100 cluster with 128 dimensions for each image.

Figure 4 - feature extraction with SIFT, which is transferred to 120\*128 array by K-mean.

* **Gradient**

Gradient is a directional change in intensity or color of image. Here we use gray image, so we should find the intensity change with different orientations. **Figure 5** shows how gradient works in X axis and Y axis. We can find X axis is better than Y axis.



Figure 5 - feature extraction with gradient by x, y axis respectively

* **Canny**

Canny is used to find edge of picture. **Figure 6** shows how it works.



Figure 6 - feature extraction with drawing outline

After feature extraction, the images have been normalized and saved in a matrix with its corresponding label. Then the matrix is persistent into two npy files, called x.npy and y.npy. Thus, we don’t have to pre-process image when training model.

1. **Model design**

Since there are 200\*200\*327=13,080,000 pixels totally and unknown noise in the pre-processed images, PCA was executed to figure out major components. **Figure 7.1** shows 55% information can be explained when reaching to 100 components with gradient feature extraction.. So, we chose first 100 components as inputs for training and validation. For canny feature extraction, we didn’t use PCA for MLP since PCA plot is much more like linear, **Figure 7.2**.

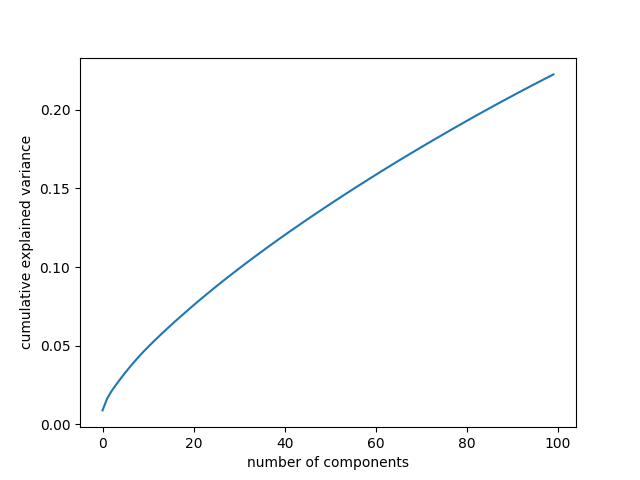
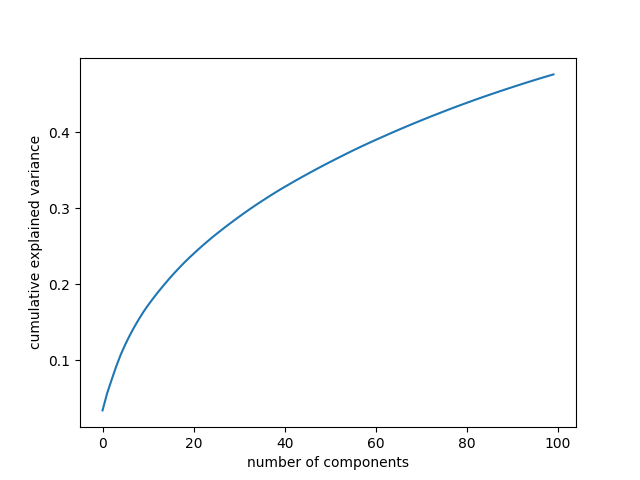


Figure 7.1 - PCA plot(Gradient) Figure 7.2 - PCA plot(Canny)

In the source database, only the last frame was labeled( total 327 frames). Lack of samples will easily lead to overfitting or bad performance in the generalization. To solve this problem, we pick up not only the last frame denoted as i, but also i -2, i - 4 frames in the same series( a series is from neutral to peak expression in one subject). **Figure 8** show the few changes in these three images.

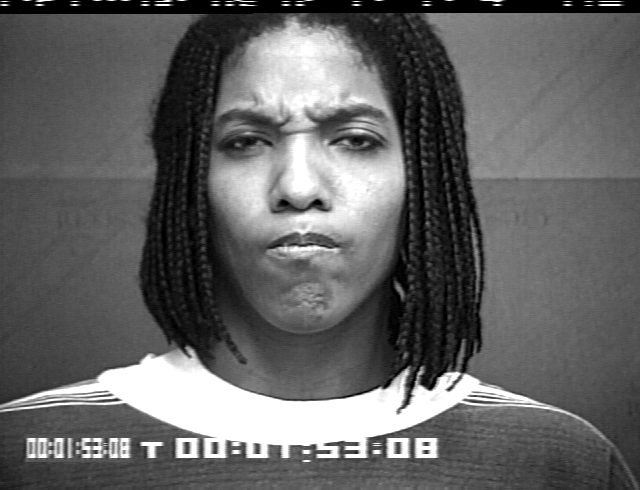
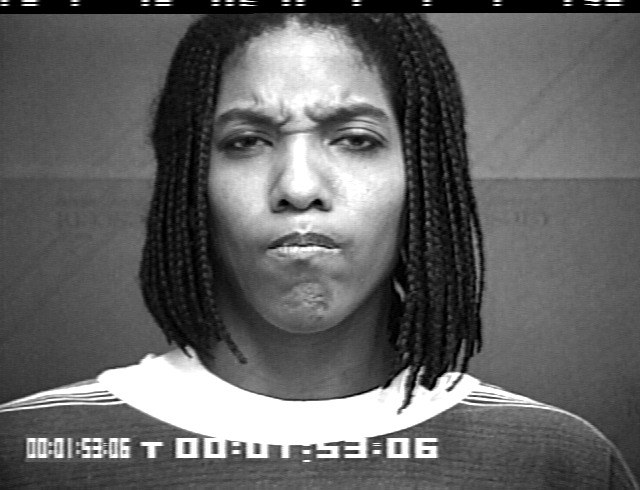
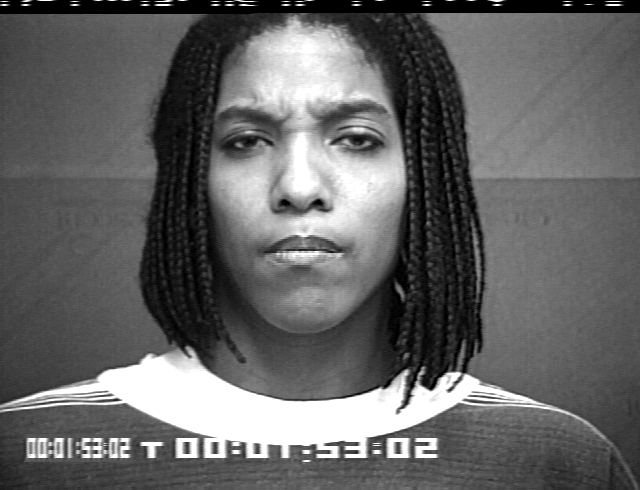


Figure 8 frown and mouth shape get few changes in these three images

For all samples, we did randomly disturbance(horizontal flip, tuning brightness and rotation 3 degree). The total sample size is increased to 327\*3=981.（Due to disturbance, few samples can not be face detected, the sample used for traning and test will less than 981)

To keep balance of training samples, stratified shuffle Split is executed which split samples into train and validation groups by label field. It guaranteed each classification has been trained in the model.

After all above steps, we construct two models: MLP and SVM. Both of them work well in the field of classification. The detail of tuning can be found in performance section.

**Performance**

Due to time limitation, we didn't tuning gradient degree also fixed PCA components to 100 as precondition for both of models.

MLP model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function | hyper parameters** | **HID(10,10)**  **ITER(500)**  **IR(0.1)**  **MM(0.5)**  **ACT(LOG)**  **SOL(SDG)**  **BAT(10)** | **HID(20,20)**  **ITER(500)**  **IR(0.1)**  **MM(0.5)**  **ACT(LOG)**  **SOL(SDG)**  **BAT(10)** | **HID(30,30)**  **ITER(500)**  **IR(0.1)**  **MM(0.5)**  **ACT(LOG)**  **SOL(SDG)**  **BAT(10)** | **HID(100,100)**  **ITER(500)**  **IR(0.01)**  **MM(0.3)**  **ACT(ReLU)**  **SOL(SDG)**  **BAT(10)** |
| SIFT |  | Acc = 21.2%  F1 =  [0.19047619  0. 0.32258065 0.06666667 0.19277108 0.09090909  0.48275862] |  | Acc = 21.2%  F1 =  [0.21686747  0. 0.27777778 0.08695652 0.2962963 0.  0.51376147] |
| Gradient - X | Acc = 79.6%  F1 = [0.64864865 0.74074074 0.82352941 0.82352941 0.94339623 0.71111111  0.87394958] | Acc = **84.5%**  F1 =  [0.6835443 0.96296296 0.94252874 0.83333333 0.98076923 0.64705882  0.86178862] | Acc = 82.9%  F1 = [0.75675676 0.82758621 0.92134831  0.8 0.96226415 0.60606061  0.83870968] | Acc = 79.1%  F1 = [0.65957447 0.91666667 0.93181818 0.77419355 0.94949495 0.64705882  0.86666667] |
| Gradient - Y | Acc = 74.7%  F1 =  [0.56470588 0.52173913 0.86956522 0.73684211 0.88888889 0.58823529  0.87603306] | Acc = 77.2%  F1 =  [0.64367816 0.75 0.86046512 0.66666667 0.8952381 0.57142857  0.87394958] | Acc = 79.3%  F1 =  [0.73170732 0.66666667 0.91954023  0.8 0.91262136 0.625  0.88709677] | Acc = 78%  F1 =  [0.6875 0.69565217 0.93181818 0.66666667 0.92929293 0.60606061  0.91666667] |
| **Function | hyper parameters** | **HID(10,10)**  **ITER(500)**  **IR(0.1)**  **MM(0.5)**  **ACT(LOG)**  **SOL(SDG)**  **BAT(10)** | **HID(50,50)**  **ITER(500)**  **IR(0.1)**  **MM(0.5)**  **ACT(LOG)**  **SOL(SDG)**  **BAT(10)** | **HID(100,100)**  **ITER(500)**  **IR(0.1)**  **MM(0.5)**  **ACT(LOG)**  **SOL(SDG)**  **BAT(10)** |  |
| Canny(NO PCA) | Acc = 63.0%  F1 =  [0.47863248 0.25 0.81481481 0.55172414 0.84536082 0.  0.93650794] | Acc = 67%  F1 =  [0.57142857 0.125 0.85714286  0.48 0.90384615 0.17391304  0.90625 ] | Acc = 61.8%  F1 =  [0.52380952 0.25 0.85365854  0.1 0.87755102 0.  0.89922481] |  |

HID = HIDDEN LAYER ITER = MAX ITER IR=LEARNING RATE

MM = MOMENTUM ACT = ACTIVATION FUNCTION SOL = SOLVER

BAT = MINI BATCH SIZE

SVM model

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Function | hyper parameters | kernal = rbf  C = 0.5 | kernal = rbf  C = 1 | kernal = rbf  C = 5 | kernal = rbf  C = 10 |
| SIFT | Not work | Not work | Not work | Not work |
| Gradient - X | Acc =75.9%  F1 = [0.79365079 0.375 0.92134831 0.1 0.94545455 0.25  0.72619048] | Acc =89.8%  F1 = [0.80597015 0.86956522 0.94382022 0.91428571 0.95412844 0.75675676  0.90769231] | Acc = **92.2%**  F1 =  [0.84375 1. 0.96629213 0.94444444 0.99047619 0.76923077  0.90076336] | Acc = 91.4%  F1 = [0.80597015 0.91666667 0.98876404 0.94444444 0.99047619 0.7027027  0.90909091] |
| Gradient - Y | Acc = 74.3%  F1 =  [0.8125 0.35294118 0.9010989 0.19047619 0.86206897 0.  0.75308642] | Acc = 84.6%  F1 =  [0.84848485 0.72727273 0.93333333 0.66666667 0.93457944 0.58064516  0.83561644] | Acc = 86.6%  F1 =  [0.82857143 0.72727273 0.90909091 0.86486486 0.9245283 0.64705882  0.88888889] | Acc = 89.8%  F1 =  [0.91176471 0.7826087 0.94382022 0.88888889 0.94339623 0.68571429  0.9037037 ] |
| Canny(components = 100) | Acc = 65.8%  F1 =  [0.68235294 0. 0.78481013 0. 0.74418605 0.  0.74482759] | Acc = 76.4%  F1 =  [0.72941176 0.31578947 0.91764706 0.4 0.85964912 0.24  0.83453237] | Acc = 72.1%  F1 =  [0.73417722  0.3 0.80952381 0.43243243 0.83928571 0.38888889  0.79032258] | Acc = 75.4%  F1 =  [0.72972973 0.5 0.80952381 0.58064516 0.86725664 0.42105263  0.828125 ] |

**Difficult & Discussion**

All the training sample are frontal face under the controllable light condition, but when we detect a side face or environment light change a bit, the accuracy will jump down rapidly.

SIFT+K-means seems doesn't work well in this project, but if we have enough time it maybe a more hopeful solution, because it is more accuracy.

68 points in Dlib library is not enough which does not capture all emotion changes of human face. If we capture more points then connect them together, we can build a simple human face polygon, then this model will not be disturbed by the brightness or noisy on face.

**Results**

In conclusion, we believe Gradient-X + Dlib + PCA + SVM is the best combo to be used for face sentiment detection. In the controlled environment, it reached up to 92.2% accuracy.