Facial Features Analysis Algorithms Comparison

Yanling He, Xin Yang
Department of Computer Science & Engineering, University of Washington

Introduction

Background:
Facial expression is a powerful communication tool in our daily life. As the computer and Internet becomes more popular today, more communications happen between the human and the machine. To understand and analyze the facial expression becomes a key point in human-computer interaction.
Many algorithms has been developed in machine learning and computer vision areas, like Principal Component Analysis (PCA), kernel method, support vector machine (SVM), boosting solution and so on. For different purposes, different algorithms may have different advantages.

Project:
Based on those backgrounds, this project focuses on implementing different face recognition and feature detection algorithms to compare the performances between them.
The project uses two major different recognition pipelines, first is using PCA to simplify the features of image data, then apply different training methods (logistic regression with 5 – fold cross validation, SVM with different kernel types). Second one is using K-NN to learn the raw image file.

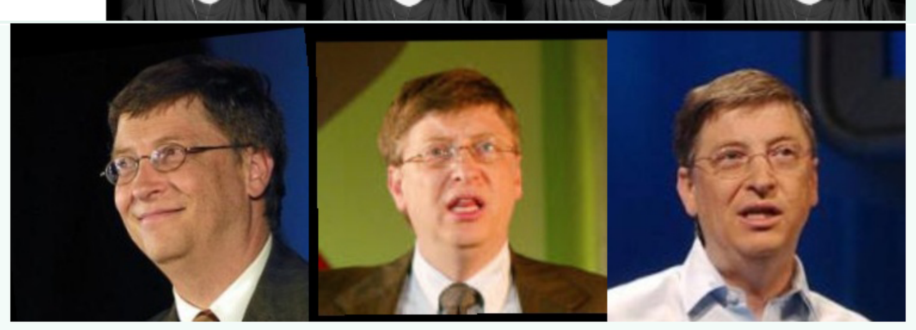
Data Processing

The input to the system is a sequence of images with faces with different facial features, like expressions, facing different positions.
The image data goes into the data processing system. It reads the labels of images and based on different features, the images are classified as an input vector y. The images are composited into an input matrix X.

Data Set

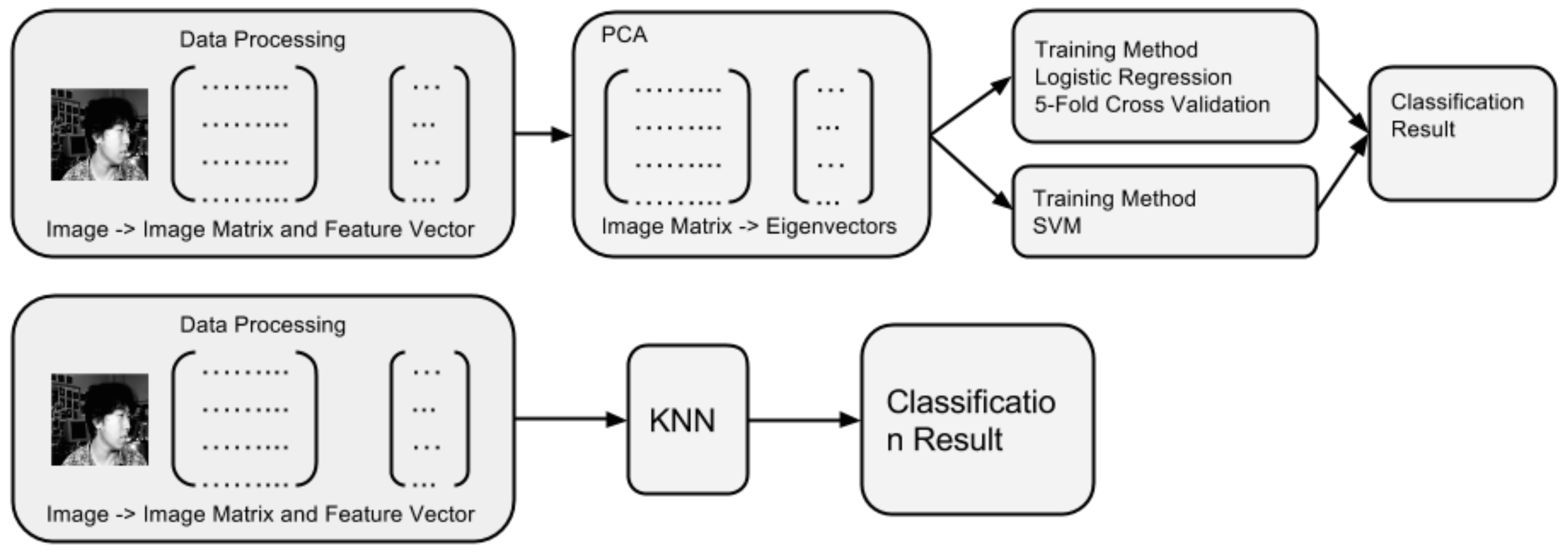
Data Set 1 — Neutral, Happy, Sad, Angry — Open / With Sunglasses

Data Set 2



Facial Features Recognition

Facial Feature Recognition Pipeline



Algorithms

PCA (Principal Component Analysis)

Decomposed image into eigenvectors:

- Calculates the covariance matrix of the input X: $\mathbf{S} = \mathbf{X}\mathbf{X}^T$
- Find the eigenvector decomposition $\mathbf{v}_i$ of S:

$$\mathbf{S}\mathbf{v}_i = \mathbf{T}\mathbf{T}^T\mathbf{v}_i = \lambda_i\mathbf{v}_i \quad \mathbf{T}\mathbf{T}^T\mathbf{T}\mathbf{u}_i = \lambda_i\mathbf{T}\mathbf{u}_i \Longrightarrow \mathbf{v}_i = \mathbf{T}\mathbf{u}_i$$

Logistic Regression

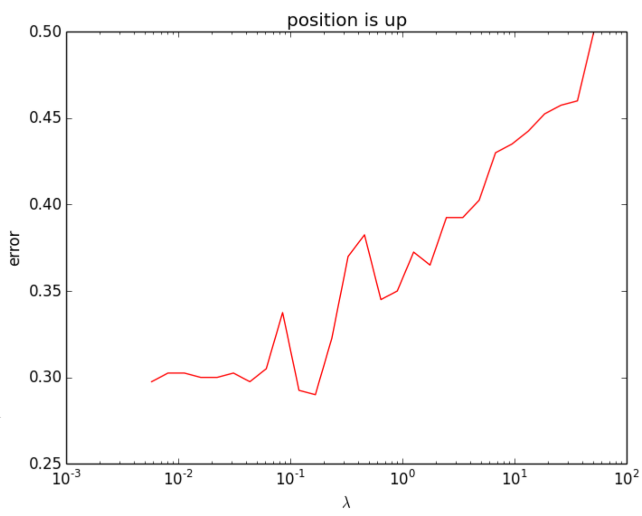
- Find coefficient w with eigenvector x:

$$\mathbf{w} = \mathbf{w} + \eta * (-\lambda * (\mathbf{w}) + \frac{1}{N} * \mathbf{X}^T \cdot (\mathbf{Y} - (1 - \frac{1}{1 + e^{w_0 + \mathbf{X}\cdot\mathbf{w}}})))$$

$$\mathbf{w_0} = \mathbf{w_0} + \eta * \frac{1}{N} * (\mathbf{Y} - (1 - \frac{1}{1 + e^{w_0 + \mathbf{X}\cdot\mathbf{w}}}))$$

5-fold Cross Validation

Bases on the w divided data into 5 groups, using 4 groups as the training data and Calculates the error of the fifth group. Same calculation on all the 5 groups to get 5 error data. Average the error data and find the $\lambda = 0.2$ with the min error.



SVM (Support Vector Machine)

SVM method modifies the loss function to ensure the solution is sparse and the predictions only depend on a subset of the training data combined with the kernel trick.
In this project we tried different kernel types to predict, including linear kernel, polynomial kernel with different degrees, sigmoid kernel and radial basis function kernel.

K-NN (K-Nearest Neighbors)

The project applies the KNN on the raw data. Using Euclidian as distance metric and based on the input neighbor number to look the nearby neighbors and predict the same output as the nearest neighbor.

$$i = argmin \parallel x^j - x^\varepsilon \parallel \qquad \hat{y} = y^i$$

Reference

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. Journal of cognitive neuroscience
Hyvarinen, A. (1999). Survey on independent component analysis. Neural computing surveys

Experiment Results

Experiment Results

Univariant Classification:

- Sunglasses Classification Image Size 32 x 30

Facing Positions	Class0 Precision	Class0 Recall	Class1 Precision	Class1 Recall
Straight	1.0	0.92857	0.93333	1.0
Up	1.0	0.75862	0.79412	1.0
Right	0.61765	0.77778	0.714286	0.53571
Left	0.84	0.75	0.78125	0.86207
Mix	0.64539	0.80531	0.73494	0.54955

- Sunglasses Classification Image Size 64 x 60

Facing Positions	Class0 Precision	Class0 Recall	Class1 Precision	Class1 Recall
Straight	1.0	0.92857	0.93333	1.0
Up	0.92857	0.89655	0.89286	0.92593
Right	0.61765	0.77778	0.71429	0.53571
Left	0.67857	0.67857	0.67857	0.67857
Mix	0.70635	0.79464	0.76289	0.66667

Multi-Classification

- Name Classification

Image Size	Logistic Regression	SVM	1NN
128 x 120	1.0	0.992	1.0
64 x 60	0.992	0.992	0.992
32 x 30	0.992	0.992	1.0

- Facing Positions Classification

Image Size	Logistic Regression	SVM	1NN
128 x 120	0.90476	0.93651	0.88889
64 x 60	0.93651	0.98127	0.88889
32 x 30	0.92063	0.95238	0.88889

Methods	Precision
Logistic Regression	0.90476
SVM [linear]	0.93651
SVM [poly, deg =3]	0.88889
SVM [sigmoid]	0.52381
SVM [rbf]	0.25397
1NN	0.88889
2NN	0.77778
3NN	0.71429

- Facial Expressions Classification

Methods	Precision
Logistic Regression	0.19048
SVM	0.23810
1NN	0.03175
3NN	0.14286

Results Analysis

Sunglasses: In sunglasses wearing classification, when face facing straight or up is more easier to predict. The error may be caused by the bad alignments of the faces and sunglasses when facing to different directions and the area of the sunglasses region reduced, so it becomes harder to detect.
Name: The classification on name prediction is very strong, especially using near neighbor algorithm. Because even people have different facial expressions, the major parts of the face remain the same.
Facing position: SVM performs better. Linear kernel is better than other kernel and less near neighbor is better than more neighbors.
Facial expression: The prediction on the facial expression is bad. Because the expressions are some small areas features. Also, by manually check the images, we found some expression are hard to tell with human eyes.