Indian Institute of Technology, Kharagpur

Machine Intelligence and Expert Systems
Term Project - Autumn 2015

# Impostor Detection with Keystroke Dynamics & Emotion Analysis from Visual Data

Vikram Mohanty - 12EC35008
Naman Mitruka - 12EC35020
Amrut Rajkarne - 12EC35017
Ankit Bansal - 12EC35024
Raunak Dhaniwala - 12EC10046

Dept. of Electronics and Electrical Communication Engineering
IIT Kharagpur

# *Acknowledgements*

The outcome of this project would not have been possible without the able guidance of **Prof. S.K.Mukhopadhyay**, who sparked our interest and enthusiasm in this domain and helped motivate and guide with lectures on Machine Learning that complemented our own learning process.
We are also deeply indebted to **Mr. Vikrant Karale**, our Teaching Assistant, whose inputs and resources have gone a long way in giving the right shape to our work.

# Contents

# List of Tables

# Abstract

Digital Security has always been a big concern with the growing advent of technology, and a boom in digital data. Numerous security measures for identifying impostors are making their way into the market, with a majority of them tapping unique identity parameters, such as fingerprints, heart rhythm, retinal scan, etc among many others, as access gateways. One such parameter is a person's keyboard typing rhythm, which is usually unique to the person in concern.

The aim of this project is to use different Machine Learning algorithms for correctly identifying an user from his typing pattern data. A keylogger was used to generate a dataset of over 10000 keystrokes per user (5 users in our case), from where certain characteristic features, which are clear indications of an user's typing pattern, were extracted to be used as our training data. A Mixture of Gaussians Model was used for classifying the users from the typing data. For the evaluation, accuracy in predictions was tested using a cross-validation dataset from 25% of the training data, while fitting the learning algorithm with the remainining 75% of the dataset. Since, this is an unsupervised learning technique, the accuracy in predictions was also tested on the training dataset itself. Further, a different test dataset was also used for the evaluation.

This project was further extended for analyzing different emotions from visual data of the human face. The JAFFE database, which is tagged with emotion labels in each image, was used for extracting the facial features, that change when a person emotes. Different existing classifiers and models were used for extracting these attributes. These attributes were then further trained with a Naive Bayes Classifier to learn the different emotions. For evaluation, a cross validation data set from 25% of the training dataset, was used.

A highest accuracy percentage of 94.78 was achieved using the Mixture of Gaussians Model on the keystrokes dataset.

# Keystroke Dynamics

## 1.1 Introduction

Keystroke dynamics is the study of whether people can be distinguished by their typing rhythms, much like handwriting is used to identify the author of a written text. Possible applications include acting as an electronic fingerprint, or in an access-control mechanism. A digital fingerprint would tie a person to a computer-based crime in the same manner that a physical fingerprint ties a person to the scene of a physical crime. Access control could incorporate keystroke dynamics both by requiring a legitimate user to type a password with the correct rhythm, and by continually authenticating that user while they type on the keyboard. Keystroke dynamics is a behavioral biometric, this means that the biometric factor is 'something you do'.

## 1.2 Methods

The methods for collecting the data, extracting the features and building the classifier are described in this section.

### 1.2.1 Data Acquisition

A keylogger was used to collect the keystroke data for all the group members. This was collected in the form of :

- "Key Event" - whether a button was pressed or released.
- "Key Code" - which button was pressed.

- "Shift","Alt","Control" - if these buttons were pressed or not.
- "Timestamp" - the exact timestamp of the event.

### 1.2.2 Feature Extraction

A custom-built feature extractor was used in the pipeline for extracting a set of 18 features from the keylogger data. These features are characteristic of an user's typing pattern. The keyboard was divided into two parts as per the convention in (**Figure 1.1**).

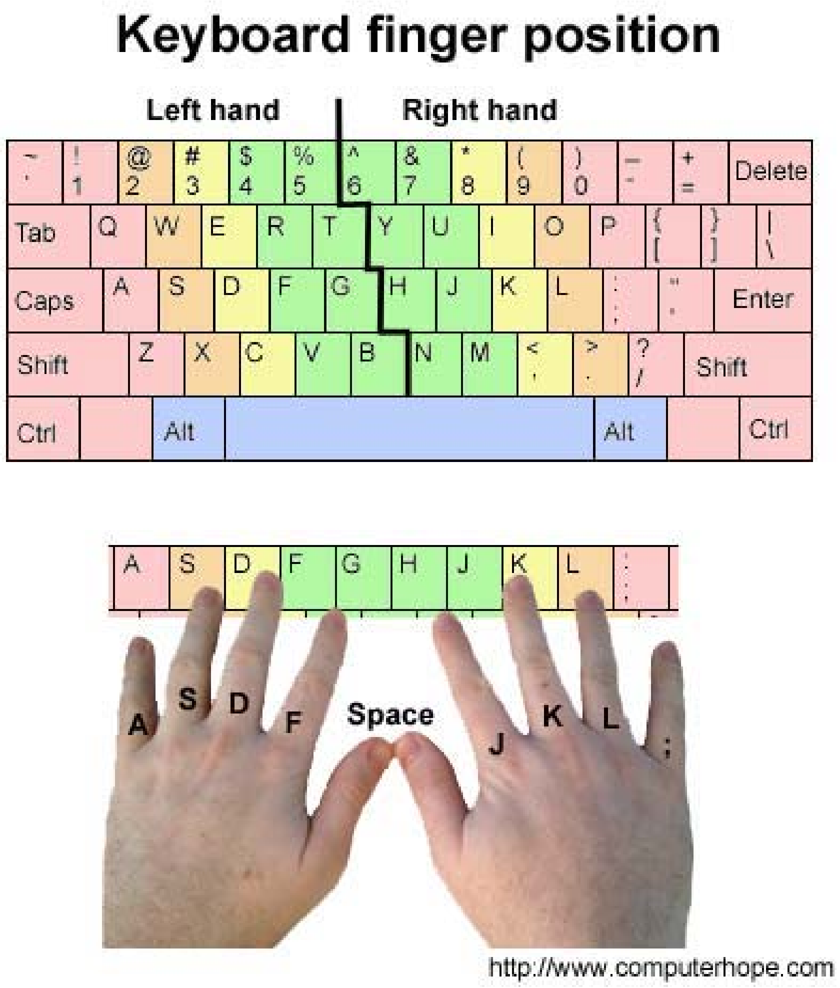

FIGURE 1.1: Keyboard with illustration of the hand's position

The extracted features can be classified into four different categories :

- **Latency**: It is defined as the time interval between **"KEY DOWN"** event (pressing) of two consecutive key strokes.
- **Hold Time**: It is the time interval for which a particular key is pressed.
- **Counts per Character**: It is a fraction of the number of times a key is pressed over the total number of keys pressed in a keystroke data.
- **Characters per Minute**: It is the average number of keys pressed in a time interval of one minute.

The difference between timestamps aided in providing latency and hold times that were used in the training data. This processed data was further used to create bins of typing sessions of the user. Each bin represents a typing session of the user. A bin is created when there is no "Key Event" recorded for a period of 10 seconds or when the user types uninterruptedly for a period of 1 minute without a break lasting more than 10 seconds. Each bin was analysed for the latency and holds times for certain combination of keys. The latency and hold times have a threshold of 1.5 seconds above which they are ignored. The mean values of the latency and hold times of a bin for the combination of keys are considered as an observation vector. A bin is considered as a valid bin if it contains more than 100 events within it, barring which it is discarded and the next bin is constructed. Additional features considered were the no. of backspaces per character of the user. This gives a sense of how prone to typing mistakes a user is and reflects mood of the user, although testing for this feature gives inconsistent results as this feature may also reflect the current nature of the text being typed and varied widely for the same user.
Another feature used is the cpm(characters per minute), which reflects the typing speed of a user and which gives a distinction among several users. The cpm feature was considered in only those bins where the user typed more than 50 valid characters so as to give a true sense of a user's typing speed.
The extracted features are shown in the **Table 1.1**.

| | | |
|---|---|---|
| Latency | lr | Transition from left hand to right hand, without shift |
| | rl | Transition from right hand to left hand, without shift |
| | Lr | Transition from left hand with shift to right hand without shift |
| | Rl | Transition from right hand with shift to left hand without shift |
| | Ll | Transition from left hand with shift to left hand without shift |
| | Rr | Transition from right hand with shift to right hand without shift |
| | ll | Transition from left hand,to left hand without shift |
| | rr | Transition from right hand,to right hand without shift |
| Hold Time | l | Hold time of keys at left part without shift |
| | r | Hold time of keys at right part without shift |
| | L | Hold time of keys at left part with shift |
| | R | Hold time of keys at right part with shift |
| | SPACE | Hold time of space key |
| | ENTER | Hold time of Enter key |
| Characters per minute | CPM | Average number of keys pressed by the user in one minute |

Table 1.1: Features extracted from the Keylogger Data

### 1.2.3 Feature Engineering

For getting better output, the attributes and the data were pre-processed. First, the unnecessary keys captured by the Keylogger, such as keys used for navigation, were filtered out as they did not have any significance in determining an user's typing pattern. Any missing values of an attribute for an observation bin are filled in by the mean of the corresponding features from the other bins. (*Filling in the missing values with the median results in a lower training and test accuracy)
The data collected in this format gives us an observation vector wherein the user continuously types a stream of text, as opposed to the method of collecting latency data of character combinations at different time stamps and including them in a single observation vector. The former proposed method which gives a more accurate and intuitive sense of the typing pattern of an user.

## 1.3 Classifier

A **Mixture of Gaussians model (GMM)** was used for classifying the obtained training data from the feature extraction step. A Gaussian Mixture Model is a parametric probability density function represented as a weighted sum of Gaussian component densities.

$$Pr(x|\Theta) = \sum_{k=1}^{K} \lambda_k Norm_x[\mu_x, \Sigma_x]$$

where $\mathbf{x}$ is predicted using the parameters $\theta = \{\mu_k, \Sigma_k, \lambda_k\}_{k=1}^{K}$. Here, $\mu_{1...K}$ and $\Sigma_{1...K}$ are the means and covariances of the normal distributions, and $\lambda_{1...K}$ are positive valued weights that sum to one.

The parameters $\theta$ are usually learned from the the training data $\{x\}_{i=1}^{I}$ using the **Expectation Maximization** algorithm, where a lower bound function is approximated to fit the GMM model over some iterations. This method can be considered analogous to K-Means clustering in terms of clusters formed over iterations.

GMMs are commonly used as a parametric model of the probability distribution of continuous measurements or features in a biometric system, such as vocal-tract related spectral features in a speaker recognition system.

The GMM Estimator provided by the **scikit-learn** library in Python was used in this project. The means of all the attributes per output label class, was provided as an initialization parameter for the model. This mean, along with the covariances and the weights, were improved over subsequent iterations. Evaluation was done using 4 different types of covariances - Spherical, Diagonal, Full and Tied.

An example is shown in the **Figure 1.2**

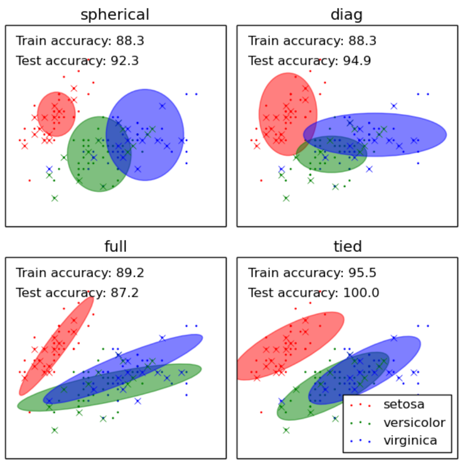

FIGURE 1.2: A GMM cluster example with output classes and plotted with 2 dimensions

## 1.4 Results

The accuracy in predictions by the Mixture of Gaussian models has been shown in the following Tables 1.2 - **??**.

| | **Tied(%)** | **Diagonal(%)** | **Spherical(%)** | **Full(%)** |
|---|---|---|---|---|
| **Training Dataset** | 86.29 | 79.44 | 43.15 | 42.83 |
| **Test-Raunak** | 70.37 | 85.19 | 92.59 | 88.89 |
| **Test-Amrut** | 94.74 | 94.74 | 78.95 | 100.00 |
| **Test-Vikram** | 77.78 | 100.00 | 11.11 | - |
| **Test-Ankit** | 89.29 | 53.57 | 25.00 | - |
| **Test-Naman** | 60.00 | 60.00 | - | - |

TABLE 1.2: Prediction Accuracy of GMM with different types of covariane matrix

In Table 1.2, it can be observed that the **Diagonal** and **Tied** type Covariance Matrix performs better than the other types. For the test datasets, 4 out of 5 users were predicted with an accuracy of over 70%. Each individual test datasets were of the length of a standard news article.

| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| Training | 87.65 | 86.29 | 89.41 | 90.27 |
| Test Raunak | - | 70.37 | 88.89 | - |
| Test Amrut | - | 94.74 | 100 | 100.00 |
| Test Vikram | - | 77.78 | - | 22.22 |
| Test Ankit | 7.14 | 89.29 | 92.86 | 92.86 |
| Test Naman | 20.00 | 60.00 | 20 | - |

TABLE 1.3: Keystrokes Dynamics Accuracy Table

In Table 1.3, the final model trained used mean to fill up missing values in bins, didn't use no.of backspace per character as a feature and considered bins with more than 100 events and latency and hold time thresholds of 1.5 s.

1. Using median to fill up missing values of bins
2. Final Model
3. Considering no. of backspaces per character as a feature
4. Considering bins with any number of events and latency and hold time thresholds of 2s
5. Using no. of backspaces per character as a feature and 4 additional features: lR,rL,rR,lL

# Emotion Analysis

## 2.1 Introduction

Emotions in human beings are usually the characteristics of the face, which shows visible change in the form of change in muscle positions, shape of the lips, eyebrows, chin and the nose. These features, if observed from an image of the human face, can be used for predicting the emotion of a person.

## 2.2 Methods

### 2.2.1 Data Acquisition

The JAFFE Database, readily available online, had a wide range of images, each labelled with the emotion of the person in the image. There were 7 different expressions - **Angry, Sad, Surprised, Happy, Fear, Disgust** and **No Expression**. These images were then passed through the feature extractor pipeline.

### 2.2.2 Feature Extractor

**Haar Classifiers**, a learning based approach where a cascade function is trained from a lot of positive and negative images, and then used to detect objects in other images. were used to tap a Region of Interest on the Face, which were further used in building Active Shape Models on the Face. **Active shape models (ASMs)** are statistical models of the shape of objects which iteratively deform to fit to an example of the object in a new image.

The combination of the classifiers and the ASMs resulted in a wide range of feature points spread across the face, tracing the eyes, nose, mouth and the face circumference (**Figure 2.3**). Using just the Haar classifiers also gave us bounding boxes around the mouth and eyes.

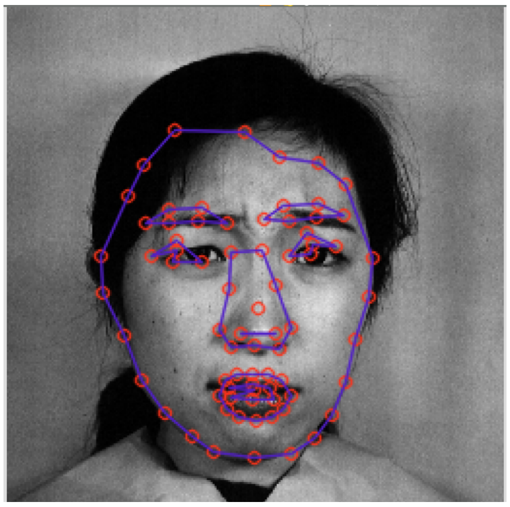

Figure 2.3: Facial Features using Active Shape Models

The use of Haar Classifiers inbuilt which detects and represents the face, eyes and mouth of any individual with a square, circle and ellipse was initially used to extract following features :

- Distance between the two eyes
- Width of mouth
- Height of mouth

Since the number of features detected were few, it was not possible to accurately differentiate between different expressions. Therefore, **Active Shape Models** were used to increase the number of points on the face to 75. From these points traced all around the face and the bounding boxes around the mouth and eyes, different features were constructed which show visible change on the face during the above mentioned emotions. These are mentioned in the Table 2.4.

| **Features** | |
|---|---|
| Chin Circumference | Eyebrow Centroid Distance |
| Chin Extreme Distance | Eyebrow Near Extreme |
| Eyebrow Far Extreme | Average Left Right Eye Distance |
| Average Top Bottom Eye Distance | Right Eyebrow Eyes |
| Left Eyebrow Eyes | Mouth Side Extreme |
| Mouth Nose Distance | Nose Chin Distance |
| Mouth Top Extreme | Eye Width - 1 |
| Eye Radius - 1 | Eye Radius - 2 |
| Eye Height - 1 | Eye Height - 2 |
| Eye Width - 2 | |
| Mouth Width | |

TABLE 2.4: Different Facial Features

### 2.2.3 Feature Engineering

The results obtained for **Angry** and **Disgust** had similar features, and therefore predicting the expression accurately was tough, and hence these two expressions were combined. No Expression and Disgust data was ambiguous, so finally 4 major expressions were considered, which have distinctive features. These expressions are: Happy, Sad, Surprise and Angry.

## 2.3 Classifier

A Naive-Bayes classifier was used for classifying the training data into different emotions. Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes theorem with the "naive" assumption of independence between every pair of features. Given a class variable and a dependent feature vector through, Bayes theorem states the following relationship:

$$P\left(y|x_1,..x_n\right) = \frac{P\left(y\right)P\left(x_1,...x_n|y\right)}{P\left(x_1,...x_n\right)}$$

Using the naive independence assumption that :

$$P\left(x_i|y,x_1,...x_{i-1},x_{i+1},...x_n\right) = P\left(x_i|y\right)$$

For all, this relationship is simplified to :

$$P\left(y|x_1,...x_n\right) = \frac{P\left(y\right)\prod_{i=1}^{n} P\left(x_i|y\right)}{P\left(x_1,...x_n\right)}$$

Since $i$ is constant given the input, we can use the following classification rule:

$$P\left(y|x_1,...x_n\right) \propto P\left(y\right)\prod_{i=1}^{n} P\left(x_i|y\right)$$

$$\widehat{y} = \underset{y}{argmax} P\left(y\right)\prod_{i=1}^{n} P\left(x_i|y\right)$$

Maximum-a-Posteriori (MAP) estimation is used to estimate P(y) and $P(x_i|y)$. The classifier was built using the scikit-learn library of Python.

## 2.4 Results

The results for accuracy after fitting the data to the Naive Bayes classifier are shown here.

In Table 2.5, 4 classes (Anger merged with Disgust, and removed No Expression and Fear) with 12 features were considered.

In Table 2.6 all 6 classes(Anger,Fear,Disgust,No Expression, Sad,Surprised) with 12

| | Gaussian | Multinomial | Bernoulli |
|---|---|---|---|
| Training | 65.87 | 56.35 | 15.25 |
| Cross-Validation | 52 | 56 | 40 |

TABLE 2.5: Prediction Accuracy with 4 classes and 12 features

features were considered.

In Table 2.7 all 6 classes(Anger,Fear,Disgust,No Expression, Sad,Surprised) with 21

| | Gaussian | Multinomial | Bernoulli |
|---|---|---|---|
| Training | 48.88 | 38.76 | 27.53 |
| Cross-Validation | 34.29 | 20.00 | 14.29 |

TABLE 2.6: Prediction Accuracy with 6 classes and 12 features

features were considered.

| | Gaussian | Multinomial | Bernoulli |
|---|---|---|---|
| Training | 50.28 | 43.50 | 15.25 |
| Cross-Validation | 42.86 | 25.71 | 14.29 |

Table 2.7: Prediction Accuracy with 6 classes and 21 features

| | Gaussian | Multinomial | Bernoulli |
|---|---|---|---|
| Training | 68 | 64.8 | 38.4 |
| Cross-Validation | 52 | 44 | 4 |

Table 2.8: Prediction Accuracy with 4 classes and 21 features

In Table 2.8 4 classes(Anger merged with Disgust, removed No Expression and Fear) with 21 features were considered.

# Conclusion

- Different models used for keystroke dynamics like- using median for missing values or using additional features like backspace count resulted in better training accuracy however the mode which had the best test accuracy was adopted.
- Naïve Bayes Algorithm possesses a drawback of returning a particular value as output. We can make a better model by assigning weights to all the emotions instead of a particular emotion which can't be done with Naïve Bayes.